Programming In Machine Language

X1 SERIES

マシン語プログラミング入門

渡辺英行・沼倉 均 共著

MIA

X1
マシン語
プログラミング入門

まえがき

Ｘ１はＺ80ＣＰＵを使ったマシンとしては、かなり完成度の高いマシンであると言われてます。しかし、ＢＡＳＩＣプログラムでは、このマシンの素晴しい機能のほんの一部しか活かすことができません。そこで、マシン語を使ってフルにＸ１をドライブしようというわけですが、一般的にマシン語はむずかしいと信じられています。たしかにコンピュータを基礎から学んでマシン語を使おうとすれば、さまざまな予備知識が必要になってむずかしいということになるでしょう。しかし、基礎は後まわしにして、まずは実践から入っていけば、マシン語なんて単なるパズルにすぎません。

本書では、あまり基礎知識にこだわらず、実践を通してマシン語を解説していきます。この方がマシン語を短期間にマスターできるはずだからです。また、マシン語プログラムの本格的な開発ツールとしてエディタ・アセンブラの全リストを掲載しています。さらに、第５章や付録で扱っているものは、資料としても使えるので、すでにマシン語をある程度、使っている人も役立ていただけるでしょう。

1984年９月　渡辺英行　沼倉均

CONTENTS

★マシン語使用のメリットとは何か。コンピュータの概要をとらえ、基礎知識を身につける。

第1章　マシン語の基礎知識 ―― 1

1・1　マシン語の特徴 ―― 2
1・2　マシン語とBASIC ―― 5
1・3　マシン語とニモニック ―― 6
1・4　コンピュータの基本構成 ―― 8
1・4・1　CPUとメモリ ―― 8
1・4・2　CPUと入出力装置 ―― 9
1・5　コンピュータで扱う数 ―― 10
1・5・1　2進数と16進数 ―― 10
1・5・2　負数の表現 ―― 11
1・5・3　BCD表現 ―― 13
1・6　2進演算 ―― 14
1・6・1　算術演算 ―― 14
1・6・2　論理演算 ―― 17
1・6・3　シフトとローテイト ―― 19

★Z80CPUの命令の動作と使い方を詳しく解説（資料として役立つよう見やすさをポイントに記述）。

第2章　Z80のマシン語命令 ―― 25

2・1　Z80のレジスタ ―― 26
2・2　Z80のマシン語命令 ―― 31
2・2・1　アドレッシング・モード ―― 31
2・2・2　Z80の命令セット ―― 35
●データの転送・交換 ―― 36
●ブロック転送とブロック・サーチ ―― 42
●演算命令 ―― 49
●ローテイト，シフト ―― 59
●ビット操作，フラグ操作 ―― 63
●ジャンプ，コール，リターン ―― 65
●入出力命令 ―― 69
●CPUコントロール命令 ―― 72

★エディタ・アセンブラの全リストを掲載し，その使い方，アセンブラの文法などを詳解。

第3章　エディタ・アセンブラ　75

3・1　概要　76
3・2　運用法　78
3・3　エディタ　80
3・4　アセンブラ　86
3・5　ローダー　87
3・6　モニタ　89
3・7　アセンブラの文法　91
●エディタ・アセンブラ　100

★個々の命令の組み合わせによる定石としてのテクニックとプログラム（具体例）による実践。

第4章　マシン語プログラミングの定石と実践テクニック　117

4・1　プロローグ　118
4・2　定石　120
4・2・1　レジスタの内容を交換する　120
4・2・2　大小比較　121
4・2・3　フィル・メモリ　124
4・2・4　ループ　125
4・2・5　F，SP，PCの値を得る　126
4・3　実践テクニック　128
4・3・1　パラメータの渡し方　128
4・3・2　ジャンプ・テーブル　131
4・3・3　文字列サーチ　132
4・3・4　乗除算　134

★サンプル・プログラムと図表を多用して，X1のIOCSとI/Oポートを解説。

第5章　IOCSとI/Oポート ―― 143

5・1　IOCS ―― 144
- キーボードからの入力 ―― 145
- 画面，プリンタへの出力 ―― 148
- 表示のコントロール ―― 153
- カセット・コントロール ―― 159
- PSG ―― 171
- PCG ―― 172
- サブCPUとの通信 ―― 175
- モニタ内サブルーチン ―― 179

5・2　ワーク・エリア ―― 183
5・3　I/Oポート ―― 195
5・3・1　シングルアクセス・モード ―― 197
5・3・2　同時アクセス・モード ―― 197
5・3・3　テキスト画面 ―― 198
5・3・4　グラフィック画面 ―― 200
5・3・5　パレット機能 ―― 202
5・3・6　プライオリティ機能 ―― 204
5・3・7　PSG(AY-3-8910) ―― 207

APPENDIX ―― 213

用語解説

本文中で，用語の右上に白ヌキの数字が付いているものは，章の終わりで解説してあります。

（例）　ビット❶

第1章

マシン語の基礎知識

1-1　マシン語の特徴
1-2　マシン語とBASIC
1-3　マシン語とニモニック
1-4　コンピュータの基本構成
1-5　コンピュータで扱う数
1-6　2進演算

1-1 マシン語の特徴

マシン語にチャレンジする前に，何のためにマシン語を使わなければならないか，ということを明らかにしておきましょう。BASICと比べた場合，マシン語を利用することで次のような効果が期待できるのです。

[1]速度の向上

マシン語を利用する最大の理由は,「速度の向上」にあります。BASICと比較した場合，数倍から数百倍高速になります。

ちょっと実験してみましょう。**リスト1-1**と**リスト1-2**はC000番地からCFFF番地までのメモリに1を書き込むものです。**リスト1-1**はBASICプログラムで実行時間は13秒，**リスト1-2**は速すぎるので100回同じこと（1180～1200行）を繰り返して実行時間は2秒しかかかりません。つまり，**リスト1-2**は650倍高速に動いていることになります。この**リスト1-2**の1090～1140行のデータがマシン語です。

リスト1-1

```
1000 '
1010 '  LIST 1-1
1020 '
1030 CLEAR &HC000
1040 TIME$="00:00:00"
1050 FOR I=&HC000 TO &HCFFF
1060   POKE I,1
1070 NEXT
1080 PRINT TIME$
1090 END
```

リスト1-2

```
1000 '
1010 '  LIST 1-2
1020 '
1030 CLEAR &HB000
1040 FOR I=&HB000 TO &HB00D
1050   READ A$
1060   POKE I,VAL("&H"+A$)
1070 NEXT
1080 '
1090 DATA 21,00,C0
1100 DATA 11,01,C0
1110 DATA 01,FF,0F
1120 DATA 36,01
1130 DATA ED,B0
1140 DATA C9
1150 '
1160 DEF USR=&HB000
1170 TIME$="00:00:00"
1180 FOR I=1 TO 100
1190   A=USR(0)
1200 NEXT
1210 PRINT TIME$
1220 END
```

②きめ細かなプログラミングが可能

マシン語によるプログラミングは，ハードウェアの機能を生かした「きめ細かな処理」を行うことが可能です。X1のBASIC (HuBASIC) は，他のBASICに比べて，かなり機能が豊富ですが，それでも割り込みを活用したプログラムなどはつくれません。結論として「BASICプログラムでできることはすべてマシン語でできるが，逆は成り立たない」と言えるのです。

③使用メモリの節約

プログラムの内容にもよりますが，一般的にマシン語のプログラムは，BASICプログラムに比べて使用するメモリが少なくて済むという利点があります。

一方，マシン語は良いことずくめ，つまり長所ばかりか，というとそうではありません。もちろん短所もあるのです。

①プログラミングが面倒

BASICの1ステップとマシン語の1ステップでは，機能的にかなり差があります。BASICの1ステップはかなりの処理能力を持っていますが，マシン語の1ステップは非常に〝原始的〟で同じ処理をしようとすると，数ステップから数10ステップにもなってしまいます。

また，システムについて(とくに入出力関係について)の知識がある程度，要求されます。たとえば，

```
PRINT  10*10
```

というプログラムをマシン語で書こうとすれば，かけ算のルーチン（マシン語にはかけ算の命令がない）と表示するルーチンが必要となるでしょう。このうち，かけ算は加算命令やシフト命令などを使うとできますが，表示ルーチンはIOCS❶またはテキスト画面❷について調べておく必要があります。

さらに，マシン語のプログラムは通常ニモニックとい

う記号で書かれますが，これをマシン語に変換しなければなりません。この作業を人間がやるとやや軽蔑的な意味を含めて，ハンド・アセンブルと呼ばれています。逆にこの作業をコンピュータに行わせるプログラムをアセンブラといいます。ハンド・アセンブルは非能率的なうえ，間違いも多いので短いプログラムでない限り使われません。結局，マシン語でプログラムをつくろうと思えば，アセンブラが必要になり，その使い方を憶えなければならないというわけです。

2デバッグに時間がかかる

前述したように，マシン語は1ステップの機能が低く，どうしてもプログラムは長くなりがちです。こうなると当然，デバッグがしづらくなります。

いくつかの例をあげましたが，マシン語でプログラミングするには，かなりの労力が必要です。しかし，その高速性は，これらの欠点を差し引いても十分"オツリ"がくるほど魅力的なものです。それゆえ，マシン語とくれば，すぐゲーム・プログラムに結びつける人が多いのですが，それは短絡的な連想ではないのです。ゲーム・センターにあるようなゲームをつくろうとすれば，マシン語を使わざるを得ないでしょう。

マシン語はすべてのコンピュータ言語のエッセンスともいえます。ですから，将来コンピュータ関係の仕事をしたいと思っている人やBASIC以外の高級言語をマスターしたいと思っている人は，ぜひマシン語にトライしてほしいものです。気づかないところで役立つことが多いでしょう。

1-2　マシン語とBASIC

マシン語は「CPUが理解ができる」唯一の言語です。もっと単純にいえば，CPUが実行できるのはマシン語だけです。「それならBASICプログラムはなぜ動くのか」，という疑問が生じますが，これはBASICインタープリタというプログラムがあるからです。インタープリタは，もちろんマシン語で書かれていますが，この働きはBASICのプログラムをひとつひとつ解析して，これと同等の処理を行うマシン語を実行するにすぎないのです。ですから，BASICプログラムはインタープリタに指示を与えるデータと見なすこともできます。

マシン語が直接CPUが実行できるのに対し，BASICプログラムはいちいち解析しなければ実行できません。処理速度に差が出てくるナゾはそこにあるのです。

1-3 マシン語とニモニック

X1にHuBASICをロードして,

```
MON ↵
```

としてモニタ・モードにしてください（↵はリターン・キーを示す)。画面にアスタリスク(*)が表示され，モニタのコマンド待ちになります。ここで,

```
D0000 ↵
```

と入力します。すると**図1-1**のように画面に表示されるでしょう。ここで2桁で表示されたものがマシン語の姿です。よく見ると，これらは0～9の数字とA～Fの英大文字で表わされています。これは16進数（後述）というマシン語の表現方法のひとつです。

図1-1 メモリのダンプ例

```
:0000=C3 FA 00 C3 7C 01 50 50 /ﾃ .ﾃ¦.PP
:0008=C3 D3 03 C3 83 04 00 18 /ﾃﾓ.ﾃ▬...
:0010=C3 D3 03 C3 BC 04 00 18 /ﾃﾓ.ﾃｼ...
:0018=C3 D3 03 C3 9D 02 00 4F /ﾃﾓ.ﾃ...O
:0020=C3 D3 03 C3 07 0E 07 20 /ﾃﾓ.ﾃ...
:0028=C3 D3 03 C3 AA 0A 00 FF /ﾃﾓ.ﾃｪ..
:0030=C3 D3 03 C3 CA 0A 00 00 /ﾃﾓ.ﾃﾊ...
:0038=C3 D3 03 C3 75 0B C3 79 /ﾃﾓ.ﾃu.ﾃy
:0040=0B C3 9A 0B C3 9E 0B C3 /.ﾃ.ﾃ.ﾃ
:0048=AE 0B C3 30 03 C3 88 09 /ｮ.ﾃ0.ﾃ▎.
:0050=00 00 46 03 D3 03 D3 03 /..F.ﾓ.ﾓ.
:0058=D3 03 D3 03 D3 03 D3 03 /ﾓ.ﾓ.ﾓ.ﾓ.
:0060=D3 03 D3 03 D3 03 C3 FA /ﾓ.ﾓ.ﾓ.ﾃ
:0068=00 63 0E 63 0E 8B 07 1E /.c.c.▌..
:0070=07 63 0E F1 07 A8 07 F7 /.c.円.ｨ.秒
:0078=07 14 08 A1 08 1B 07 BF /...｡...ｿ
```

「わずらわしい」との印象を抱く人もあるでしょう。心配はいりません。マシン語でプログラムをつくるといっても16進数で記述するわけではありません。この16進数で表わされるマシン語をもう少し人間にわかりやすくするために考えられた記号がニモニックです。たとえば,

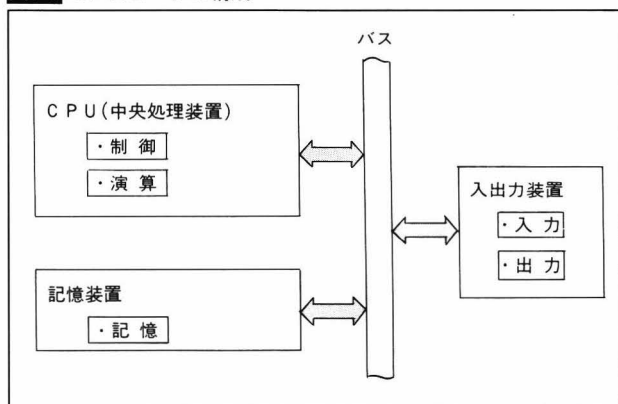
Aレジスタの内容とBレジスタの内容を足して，その結果をAレジスタに入れるといった場合（これをBASIC風に書くとA＝A＋Bとなる），16進数で表わしたマシン語では〝80〟となりますが，ニモニックでは，

```
ADD   A, B
```

となるわけです。ADDは英語で〝加える〟という意味です。このようにニモニックは，意味のある言葉を略したものです。ニモニックで表わされる言語をアセンブリ言語と呼びます。個々のニモニックについては第2章で，アセンブラについては第3章で扱います。

1-4 コンピュータの基本構成

コンピュータと呼ばれるものは，マイコンから大型機に至るまで入力，出力，記憶，演算，制御の５つの機能を持っています。これらの構成は**図1-2**のようになっています。

図1-2 コンピュータの構成

演算と制御は，CPU (Central Processing Unit = 中央処理装置) が受け持ち，記憶装置（メモリ）はプログラムやデータを記憶します。入力や出力は外部とのデータのやりとりを受け持つもので，これにはキーボード，ディスプレイ，プリンタ，フロッピィ・ディスクなどがあります。

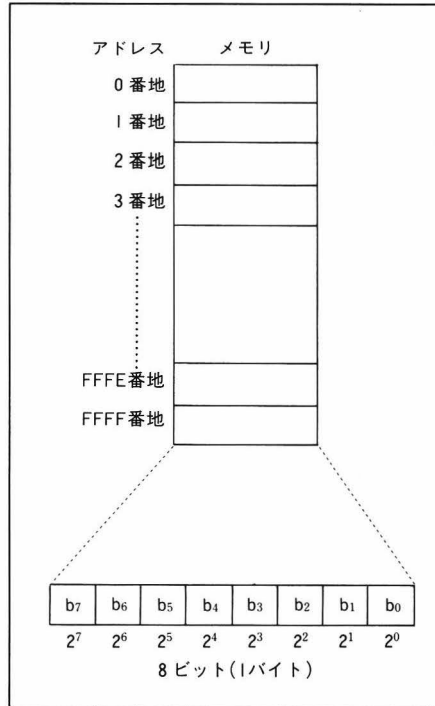
1-4-1 CPUとメモリ

CPUは，メモリに書き込まれたプログラム(マシン語)を読み込んで，それに応じた動作をします。CPUは，読み込んできたマシン語を解読する命令解読部，解読されたものにしたがってCPU全体の機能をコントロールする

コントロール部，CPU内のデータを記憶するレジスタ，さらに命令を実行した後の状態を示すフラグにわけられます。

メモリは，ビット❸の集まりでビットはバイト❹単位に区切られて，その単位でCPUから読み出し，書き込みが行われます。メモリには，バイト単位にアドレス（番地）が割り当てられており，アドレスをもとにどのメモリを参照するかを決めています。アドレスは2バイト（16ビット）で表わされ，それぞれのアドレスには1バイトのデータが入ります（図1-3）。

X1のBASIC MANUALにメモリ・マップが書かれていますが，これはメモリがどう使われているかを示すものです。

図1-3 メモリの構成

1-4-2 CPUと入出力装置

X1にはZ80❺というCPUが使われていますが，このCPUはメモリとは別に入出力用に使う256バイトの空間を持っています。この空間のことをI/Oポートまたは単にポートと呼んでいます。I/OはInput/Outputの略で入出力を意味します。ポートは〝港〟のことで，外部とデータをやりとりするときの中継点です。

Z80には入出力命令があり，これを実行するとCPUはポートを中継してデータの入出力を行います。

1-5 コンピュータで扱う数

コンピュータは，内部で電気的なON/OFFで動作しています。このON/OFFを1/0の数に置きかえたものを2進数と呼びます。2進数は1桁では，0と1の2つの状態しか表わせません。2桁になると〝00″，〝01″，〝10″，〝11″の4つの状態を表わすことができます。2進数の1桁1桁をビットと呼びます。

Z80は，8ビットのCPUですが，これは同時に8ビットのデータを扱えることを意味します。8ビットでは，0から255(2^8-1)の状態を表わせます。

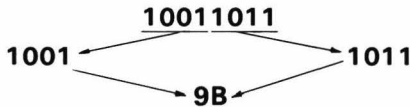
1-5-1 2進数と16進数

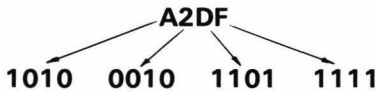
2進数は，大きな数になると桁が多くなって表示に場所をとるし，0と1ばかり並んでいたのではいくつなのかわかりづらくなります。そこで16進数がよく使われます。**図1-1**では16進数で表示されています。

表1-1 10進，2進，16進の対応

10 進 数	2 進 数	16 進 数
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

16進数は，２進数の４ビットを１桁で表現することができます。**表1-1**の対応表を見てください。２進数から16進数に変換するには，まず２進数を右端から４桁ずつに分け，それを**表1-1**を参照して変換すればできます。

〈例〉　２進数〝10011011〞を16進数に変換する。

逆に16進数を２進数に変換するには，16進数の１桁を２進数の４桁に変換していきます。

〈例〉　16進数〝A2DF〞を２進数に変換する。

これらの変換作業は，慣れてくれば表を見なくてもできるようになります。

1-5-2　負数の表現

これまで説明してきた２進数や16進数には符号がありませんでした。実際には，負数を扱う場合もあるので，どうやって表現しているか触れておきましょう。

ふだん，私たちは負数を表現するために〝－〞記号を使います。コンピュータの中では，この符号も０と１で表わします。一般に０はプラス，１はマイナスを示すために使われます。

表1-2に８ビット(1バイト)で表現できる符号つき整数を，**表1-3**に符号なし整数を示します。**表1-2**の２進数の最上位ビット（ビット７）が符号ビットです。このような数の表現法を２の補数表示といいます。

表より，２進数の11111111は符号なしだと255，符号つ

きだと－1を表わすことになります。2進数では，同じなのに符号のあるなしによってまったく違う数字になってしまいます。ある数値を符号つきで扱うか符号なしで扱うかは，プログラムによって決めます。

2の補数への変換は，まず2進数のビットを反転します。つまり，〝0〞なら〝1〞に，〝1〞なら〝0〞にします。反転した値に1を加えると2の補数になります。

〈例〉　2進数〝00000011〞を2の補数にする。

00000011（10進数の3）

↓　全ビットを反転

11111100

↓　1を加える

11111101（10進数の－3）

表1-2 2の補数表示による符号つき8ビットの値

10進数	2進数							
	2^7（符号）	2^6	2^5	2^4	2^3	2^2	2^1	2^0
＋127	0	1	1	1	1	1	1	1
＋126	0	1	1	1	1	1	1	0
＋125	0	1	1	1	1	1	0	1
・					・			
・					・			
＋64	0	1	0	0	0	0	0	0
・					・			
・					・			
＋16	0	0	0	1	0	0	0	0
・					・			
・					・			
プラス ↑ ＋3	0	0	0	0	0	0	1	1
＋2	0	0	0	0	0	0	1	0
＋1	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0
－1	1	1	1	1	1	1	1	1
↓ －2	1	1	1	1	1	1	1	0
マイナス －3	1	1	1	1	1	1	0	1
・					・			
・					・			
－16	1	1	1	1	0	0	0	0
・					・			
・					・			
－64	1	1	0	0	0	0	0	0
・					・			
・					・			
－126	1	0	0	0	0	0	1	0
－127	1	0	0	0	0	0	0	1
－128	1	0	0	0	0	0	0	0

表1-3 符号なし8ビットの値

10進数	2進数							
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
255	1	1	1	1	1	1	1	1
254	1	1	1	1	1	1	1	0
253	1	1	1	1	1	1	0	1
・					・			
・					・			
129	1	0	0	0	0	0	0	1
128	1	0	0	0	0	0	0	0
127	0	1	1	1	1	1	1	1
・					・			
・					・			
64	0	1	0	0	0	0	0	0
・					・			
・					・			
16	0	0	0	1	0	0	0	0
・					・			
・					・			
3	0	0	0	0	0	0	1	1
2	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0

1-5-3　BCD表現

BCDはBinary Coded Decimalの略で，2進化10進数と訳されます。これは，2進数によって10進数を表わすために考えられたものです。Z80にも10進数の演算のための命令があります。具体的には，4ビットの2進数で10進数の0～9を表わします。

たとえば，10進数の〝139〟はBCDで〝0001 0011 1001〟と表わします。このようにBCDは，見かけは2進数ですが基本的には10進数なので人間にとってたいへんわかりやすいものです。半面，2進数の計算に比べて時間がかかるし，同じ数を表わすのに2進数より多くのビットを使います。

このほか，浮動小数点などの表現もありますが，マシン語のプログラムではあまり使われませんし，入門の範囲を超えるので本書では扱いません。

1-6　2進演算

ここでは，コンピュータが2進数を使ってどのように演算しているかを説明します。演算には，算術演算（いわゆる四則計算），論理演算，シフトとローテイトがあります。

1-6-1 算術演算

算術演算とは，いわゆる四則計算ですから，加減乗除の4つの計算を指します。しかし，Z80には乗除算の命令がなく，いくつかの命令を組み合わせでつくります。これらは第4章で紹介しますので，ここでは加減算について述べます。

1 加算

始めに1ビットどうしの加算について考えてみましょう。

1ビットどうしの加算には次の4通りの組み合わせがあります。

```
   0      0      1      1
  +0     +1     +0     +1
 ----   ----   ----   ----
   0      1      1     10
```

1＋1＝2でないことに注意してください（2進数ですから0と1しかありません)。1＋1＝10となって桁上がりします。

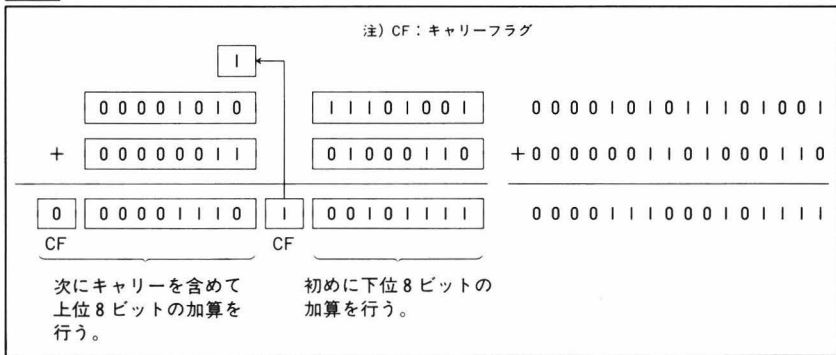
複数ビットの加算では，1ビットずつ右端から計算していけばよいのです。もし，桁上がりが発生したら次の桁では，その桁上がりも含めて加算します。別に特別なことではなく，10進数の計算と同じです。

〈例〉

```
  01110101
 +01010100
 ---------
  11001001
```

Z80では，基本的に8ビット（1バイト）で計算します。計算結果は，8ビットの値とキャリー(桁上がり)フラグ❻で求められます。8ビットどうしの加算で，桁上がりがあればキャリーフラグが1になり，桁上がりがなければキャリーフラグは0になります。たとえば，11111111＋11111111の計算を行うとキャリーフラグは1になり，00110011＋00110011ではキャリーフラグは0になります。

キャリーフラグを使うことで，8ビットより大きい数の計算を行うことができます。例を**図1-4**に示します。

図1-4 **16ビット(2バイト)の加算**

2 減算

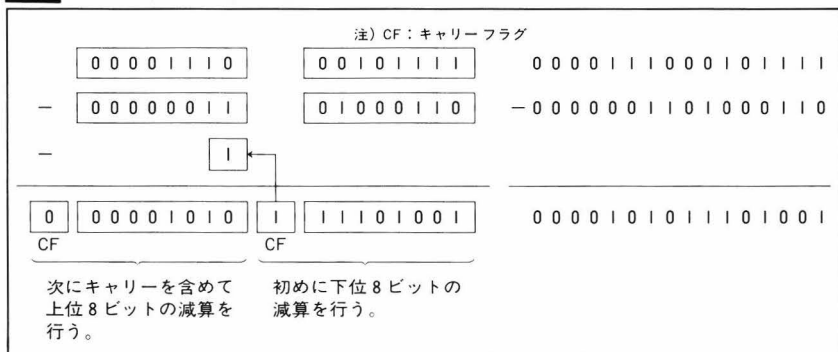
加算と同じように1ビットの2進数どうしの減算を考えてみましょう。

1ビットどうしの減算には次の4通りの組み合わせがあります。

0	1	1	10
−0	−1	−0	−1
0	0	1	1

減算も10進数の減算と同じで，引く数が大きい場合(4番目の例)，上のビットから1を借りてきて計算します。この借りをボローと呼びますが，Z80にはボローフラグ

というものがありません。キャリーフラグがボローフラグを兼用しています。したがって，キャリーフラグは，加算の後と減算の後では意味が異なることになります。加算の場合と同じように，キャリーフラグを使えば数バイトにわたる数値でも減算ができます。例を**図1-5**に示します。

図1-5 16ビット(2バイト)の減算

③ 2つの補数表示と加減算

いままでの説明は，符号なし整数についての加減算でしたが，今度は符号付き整数(2の補数表示による)の加減算について考えてみましょう。

たとえば，−13（8ビットの2進数で「11110011」)に20（8ビットの2進数で「00010100」を加算してみると，

```
  11110011
 +00010100
 ---------
  00000111  (10進数の7)
```

と，−13＋20の計算結果として7が求められます。

次に，3(2進数で「00000011」)から−5(2進数で「11111011」)を減算してみると，

```
  00000011
 -11111011
 ---------
  00001000  (10進数の8)
```

と，3−(−5)の計算結果として8が求められます。

このように，負数を2の補数で表わしておけば，符号なし2進数と同様に演算できます。

正負混合計算であっても，数が正か負かを意識せずに

計算できるのが２の補数という表現法なのです。

図1-6に８ビットの符号付き整数の計算例をいくつか示しておきます。

図1-6 2の補数表示による計算例

①	+29	…………………	00011101
	+) −96	…………………	+) 10100000
	−67	…………………	10111101
②	−100	…………………	10011100
	+) +80	…………………	+) 01010000
	−20	…………………	11101100
③	+100	…………………	01100100
	+) −100	…………………	+) 10011100
	0	…………………	00000000
④	−8	…………………	11111000
	−) −2	…………………	−) 11111110
	−6	…………………	11111010
⑤	0	…………………	00000000
	−) +1	…………………	−) 00000001
	−1	…………………	11111111
⑥	−100	…………………	10011100
	−) +10	…………………	−) 00001010
	−110	…………………	10010010

1-6-2　論理演算

論理演算で扱う値には,〝真〟と〝偽〟の２つしかありません。これは，２進数の１と０に対応するもので，コンピュータの中では真は１，偽は０として演算します。扱う値が０と１だけということで，最もコンピュータらしい演算といえます。

論理演算の基本的な演算は，NOT(否定)，AND(論理積)，OR(論理和)の３種類です。

NOTは１つの入力に対して演算を行うもので,入力が０なら１を，１なら０という結果になります。ANDやORは２つ以上の入力に対して演算を行うものです。ANDは入力のすべてが１ならば１に，ひとつでも０があれば０

になります。ORは入力のすべてが0ならば0に，ひとつでも1があると1になります。

Z80には，NOT，AND，ORの論理演算命令のほかにXOR（排他的論理和）という論理演算命令があります。XとYというそれぞれ1ビットの入力があったときのAND，OR，XORの演算結果（これを真理値表と呼ぶ）を**表1-4**に示します。表からわかるように，XORは両方同じ値のときは0，違うときに1になります。

表1-4

X　Y	X AND Y	X OR Y	X XOR Y
0　0	0	0	0
0　1	0	1	1
1　0	0	1	1
1　1	1	1	0

論理演算は8ビット（1バイト）で計算されますが，計算方法は同じ桁どうしで論理演算を行えば良いのです。例を**図1-5**に示します。

図1-5 8ビットの論理演算

```
NOT) 1 0 1 0 1 0 1 0          1 0 1 0 1 0 1 0
     ---------------     AND) 0 0 1 1 0 0 1 1
     0 1 0 1 0 1 0 1          ---------------
                              0 0 1 0 0 0 1 0

     0 1 0 1 0 1 0 1          1 0 1 0 1 0 1 0
XOR) 0 0 1 1 0 0 1 1      OR) 0 0 1 1 0 0 1 1
     ---------------          ---------------
     0 1 1 0 0 1 1 0          1 0 1 1 1 0 1 1
```

さて，論理演算は，いったいどういうときに使われるかというと，多くの場合，複数のビットをセット（1にする）したり，リセット（0にする）したり，または反転（1を0に，0を1に）したりするときに使われます。これらの例を**図1-6**，**図1-7**，**図1-8**に示します。このほかに，比較やフラグの操作などにも使われますが，これについては第2章と第4章で詳しく見ていきます。

図1-6 ORによるビットのセット

2進数〝00000110〞(10進数の6)を文字(ASCIIコード)の〝6〞(2進数の00110110)に変換する。

```
    00000110…………10進数の6
OR) 00110000…………セットしたいビットだけを1にする
    00110110…………ASCIIコードの〝6〞の文字
```

図1-7 ANDによるビットのリセット

文字(ASCIIコード)の〝9〞(2進数の00111001)を10進数の9(2進数の00001001)に変換する。

```
     00111001…………文字の〝9〞
AND) 00001111…………リセットしたいビットだけを0にする
     00001001…………10進数の9
```

図1-8 XORによるビットの反転

① 2進数の10101010の全ビットを反転する。

```
     10101010
XOR) 11111111…………反転したいビットを1にする
     01010101…………結果はNOTと同じになる
```

② 2進数の10101010を0にする。

```
     10101010
XOR) 10101010…………同じ値どうしでXORを行うと…
     00000000…………結果は0になる
```

1-6-3 シフトとローテイト

Z80には，データの内容を左右に移動（シフト）させたり，回転（ローテイト）させたりする命令があります。これらの命令を使うと1命令で8ビットをひとかたまりとして1ビットずつシフトまたはローテイトさせることができます。

1 シフト

シフトには論理シフトと算術シフトがあり，さらにその中に右へ移動するものと左へ移動するものがあります。

論理シフトは，ビットを右または左に移動させて空いたところに0を入れ，はみ出した値をキャリーフラグに入れます。この動作を図示したものが**図1-9, 1-10**です。

図1-9 論理左シフト

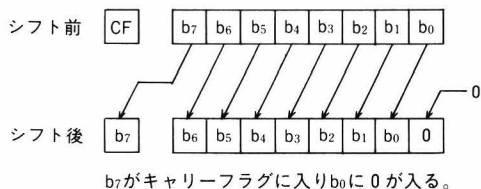
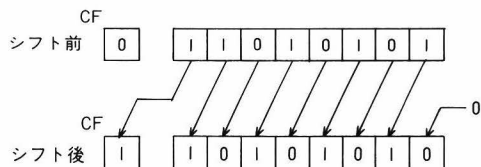



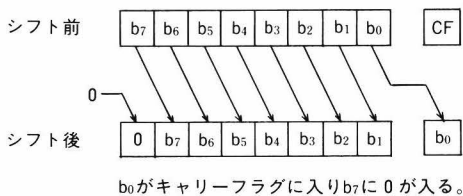
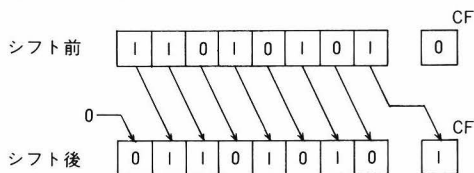
図1-10 論理右シフト

算術シフトとは，シフトによって符号ビット（第７ビット）が変化しないシフトのことです。算術右シフトの動作を**図1-11**に，算術左シフトの動作を**図1-12**に示します。

図1-11 算術右シフト

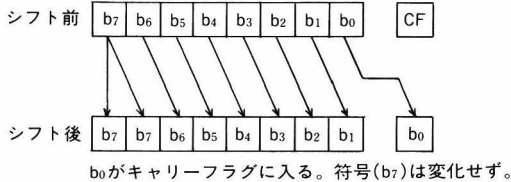
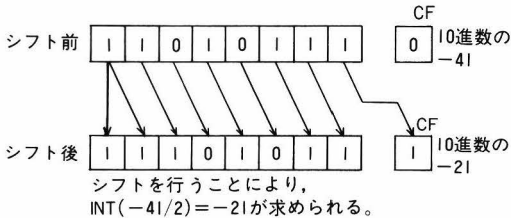
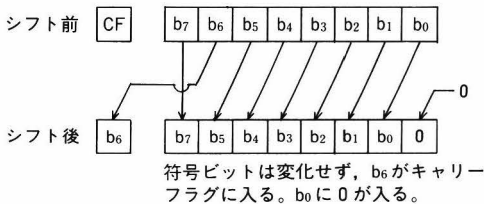
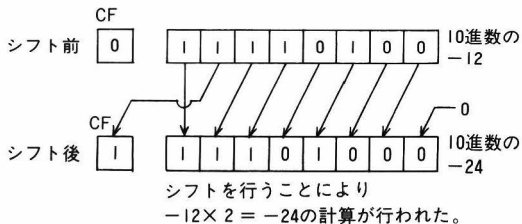



図1-12 算術左シフト

しかし，この算術左シフトに相当する動作を行う命令はありません。ただし，算術左シフトという名前の命令があり，この動作は論理左シフトと同じです。また，この論理左シフトという名前の命令はありません（このあたりは少々変な命令体系になっていますが，このことは第2章で説明します)。

2 ローテイト

ローテイトとは，左または右へビットを回転させるもので，キャリーフラグを含めて回転させるものをローテイト，キャリーフラグを含めずに回転させるものをローテイト・サーキュラと呼びます。動作を**図1-13，1-14**に示します。

シフトやローテイト命令は，乗除算の演算を行う場合によく使われます。たとえば，2進数の00000110(10進数の6）を左へ1ビット分シフトすれば，00001100（10進数の12）となり2倍になったことになります。さらに1ビット分シフトすれば，4倍になります。逆に右へ1ビット分シフトすれば，½になります。実際例は，第4章で見ていきます。

図1-13 ローテイトの動作

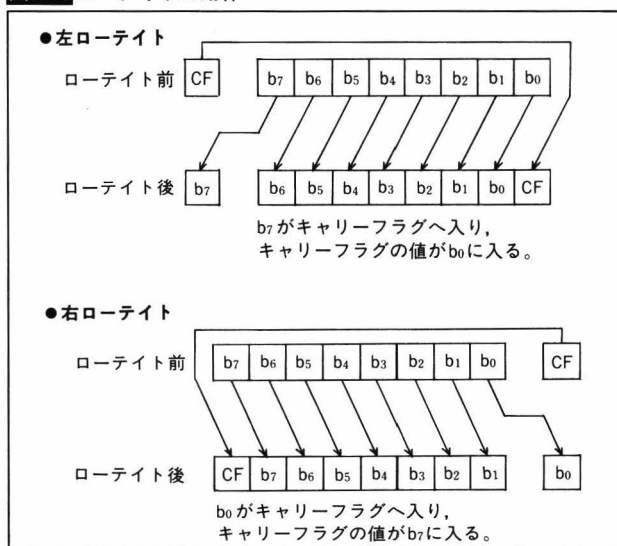
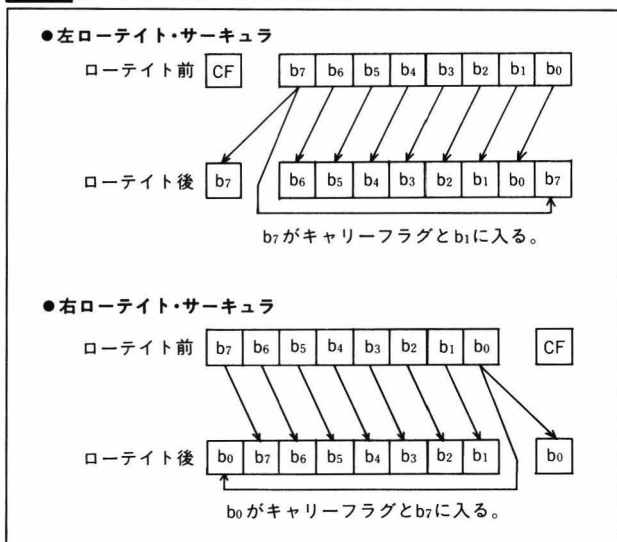



図1-14 ローテイト・サーキュラの動作

用語解説

❶IOCS

IOCSはInput/Output Control Systemの頭文字をとったもので，入出力コントロール・システムと訳される。これは，キーボードからキー入力を受け取ったり，画面に文字を表示するなどといった入出力の基本的な処理を集めたサブルーチン群である。IOCSを使えば，キー入力なども簡単にできる（逆に使わなければ，この機能に相当するルーチンをつくらなければならない。それにはハードウェアを熟知していなければならないので，より多くの労力が必要となる）。IOCSの解説は，第5章で行う。

❷テキスト画面

文字（ASCIIコード）を表示するための画面。これに対して線を引いたり円を描いたりする画面をグラフィック画面と呼ぶ。

❸ビット

0と1で表わされる2進数の1桁をビットと呼ぶ。

❹バイト

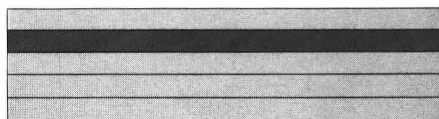
8ビットを1バイトと呼び，メモリの大きさを表わす単位としても使われる。最下位のビットをb_0と表記し，順にb_1，b_2…b_7と表わす。とくに最下位のビットをLSB（Least Significant Bit），最上位のビットをMSB（Most Significant Bit）と呼ぶ。

❺Z80

Z80は，米国ザイログ社が開発したCPUで，実行速度の違いでZ80，Z80A，Z80Bなどのファミリーがある。X1では，Z80A（4MHz）を使っているが，ソフト的には全く同じものなので，本書ではZ80と表記する。

❻キャリーフラグ

CPUのなかには演算結果の状態を記憶するために，フラグ・レジスタと呼ばれるレジスタがある。キャリーフラグのほかには，ゼロ・フラグ，サイン・フラグなどがあり，比較や演算などのときに参照される。詳しくは第2章で解説する。

第2章

Z80のマシン語命令

2-1 Z80のレジスタ
2-2 Z80のマシン語命令

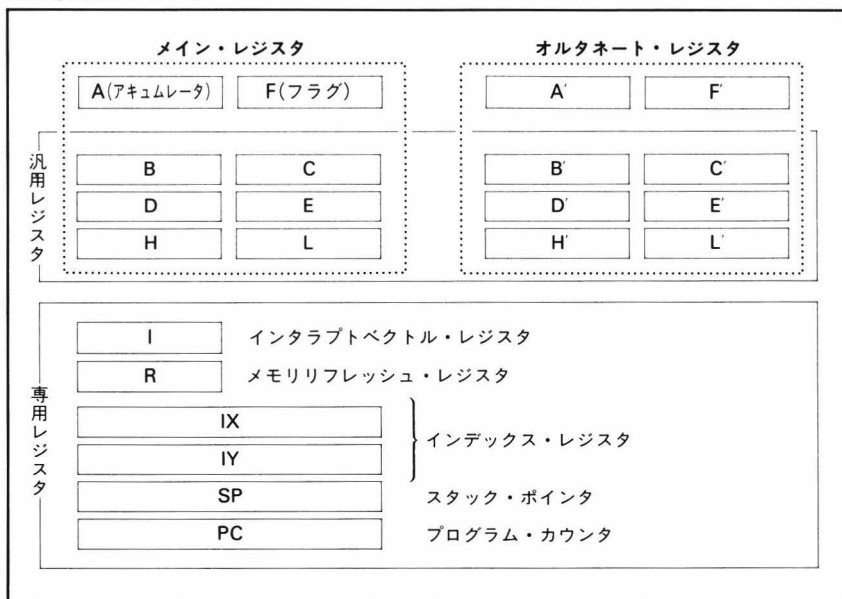
2-1 Z80のレジスタ

CPU内には,レジスタという記憶装置があり,演算を行うためのデータを記憶したり，また逆に演算結果を記憶したりします。BASICでいうと変数に相当するものですが，変数はメモリが許す限りいくらでも自由に名前を付けて使うことができるのに対し，CPUのレジスタは一定の数しかなく，また名前を付け変えることもできません。

Z80には**図2-1**に示すようなレジスタがあります。このレジスタ群は，８ビット・レジスタと16ビット・レジスタにわけられ，さらにメイン・レジスタとオルタネート・レジスタにわけられます。図では，小さな箱が８ビット(ただしRだけは７ビット)，大きな箱が16ビット・レジスタとなっています。

図2-1 Z80のレジスタ

	メイン・レジスタ		オルタネート・レジスタ	
	A(アキュムレータ)	F(フラグ)	A′	F′
汎用レジスタ	B	C	B′	C′
	D	E	D′	E′
	H	L	H′	L′

専用レジスタ		
	I	インタラプトベクトル・レジスタ
	R	メモリリフレッシュ・レジスタ
	IX	インデックス・レジスタ
	IY	インデックス・レジスタ
	SP	スタック・ポインタ
	PC	プログラム・カウンタ

1アキュムレータ(A)

アキュムレータの頭文字をとって名付けられたAレジスタは，その名のとおり演算の中心的な役割りを果たします。実際，プログラムではもっともよく使われるレジスタです。

2フラグ・レジスタ(F)

フラグ・レジスタは，演算結果の状態を記憶するためのレジスタです。このレジスタの構成を**図2-2**に示します。

●キャリーフラグ(CY)

第1章の演算の項でも登場しましたが，加算か減算かで意味が異なります。加算時には，キャリーがあれば1（セット）になり，なければ0（リセット）になります。減算時にはボローがあると1，なければ0になります。また，論理演算時は必ず0になります。

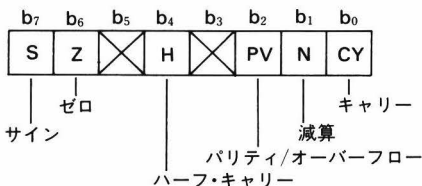
●減算フラグ(N)

加算命令を実行するとリセット（0になる）され，減算命令を実行するとセット（1になる）されます。BCD演算のときに使われます。

●パリティ／オーバーフローフラグ

このフラグもキャリーフラグと同様に2つの意味を持ちます。

論理演算を実行すると結果のパリティを示します。パリティとは，演算結果の1になっているビットの数のことで，偶数個（Even）か奇数個（Odd）かで表わします。

図2-2 フラグ・レジスタの構成

算術演算を実行すると2の補数演算の桁あふれ（オーバーフロー）を示します。演算結果の値が2の補数の範囲（－128～＋127）を超えたときセットされます。

●ハーフ・キャリーフラグ(H)

BCD演算のときに使うフラグです。BCD演算は，4ビットずつを単位として計算しますが，上位・下位4ビット間での桁あふれ，桁借りの有無を検出します。

●ゼロ・フラグ(Z)

キャリーフラグとならんで良く使われるフラグです。演算結果が0ならセットされ，0でなければリセットされます。

●サイン・フラグ(S)

演算結果の符号ビットが入ります。

③汎用レジスタ（B, C, D, E, H, L）

汎用レジスタは，8ビットで使うことも16ビットで使うこともできます。16ビットで使うときは，BC，DE，HLというペア・レジスタと呼ばれる組み合わせで使われます。

ペア・レジスタは主にメモリやアドレスの指定（ポインタ）として使うことが多いのですが，それ以外にもペア・レジスタ間で16ビットの演算（ただし加減算のみ）ができます。この場合，HLレジスタが16ビットのアキュムレータの役割りを果たします。

④オルタネート・レジスタ

これまで紹介してきたA，F，B，C，D，E，H，Lには，もう1組のレジスタ群があります。これがオルタネート・レジスタで裏レジスタとも呼ばれます。しかし，メイン・レジスタと同時に使うことはできません。オルタネート・レジスタは，メイン・レジスタと区別するためにダッシュ（′）を付けて表わします。

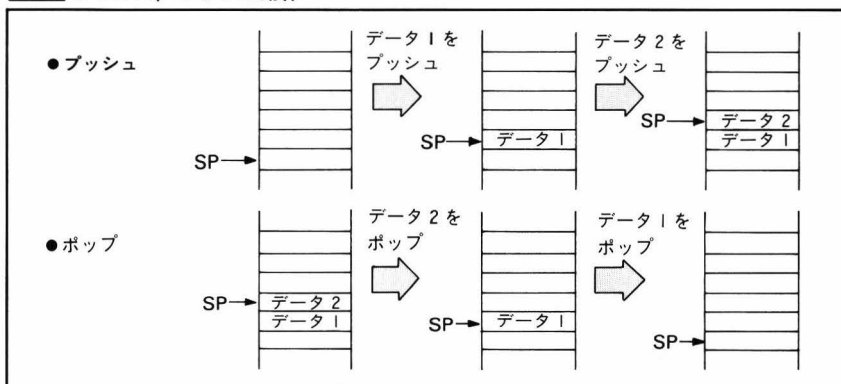
⑤プログラム・カウンタ（PC）

プログラム・カウンタは，次に実行すべきマシン語のアドレスを指しています。PCは，PCが指しているアドレスの命令を1バイト読み取るごとに自動的にインクリメント（＋1）されます❶。

⑥スタック・ポインタ

スタック・ポインタは，メモリ内にあるスタックの先頭アドレスを指すものです。スタックとは「積み重ねる」という意味で，データやアドレスを一時的に記憶するためのメモリ領域のことです。

スタックにデータやアドレスを書き込むことをプッシュ，読み出すことをポップと呼びます。Z80では，一度にプッシュ，ポップするのは16ビットとなっています。プッシュ，ポップの動作は**図2-3**のようになります。

図2-3 プッシュ，ポップの動作

⑦インデックス・レジスタ（IX, IY）

IX, IYという2本のインデックス・レジスタは，ペア・レジスタと同様にアドレスの指定に使われます。インデックス・レジスタは自身の内容を変えずに－128～＋127バイトまでのアドレスを指定することができます。

⑧インタラプトベクトル・レジスタ（I）

Z80には割り込み❷のモードが3種類あり，Iレジスタはそのうちのモード2と呼ばれる割り込みで使われます。

X１ではモード２の割り込みを使っているので，Iレジスタを操作するのは避けた方がよいでしょう。

⑨メモリリフレッシュ・レジスタ（R）

このレジスタだけは７ビット構成で，ダイナミックRAM❸のためにリフレッシュ・アドレス❹を出力します。このレジスタはゲームなどで乱数値を得るときによく使われます。

2-2 Z80のマシン語命令

マシン語命令は1バイトから4バイトの数値ですが，この数値を組み合わせてプログラムをつくるわけではありません。これには1章でも説明したように，アセンブリ言語を使います。たとえば〝Aレジスタに16進数の5Fを入れる〟という命令ならば〝LD A，5FH〟と書きます。これは〝3E，5F〟という2バイトのマシン語に対応します。ここでLDは，〝Load〟（ロードは『～へ入れる』という意味）の略で，AはAレジスタ（アキュムレータ）を示します。また，5FHのHは16進数であることを示します。

アセンブリ言語はニモニックとオペランドにわけられます。ニモニックは〝LD〟のように動作を表わし，オペランドは〝A，5FH〟のようにレジスタや数値などの指定を行います。

2-2-1 アドレッシング・モード

アドレッシング・モードとは，アドレスやレジスタの指定方法のことです。ここでは，LD命令を例にアドレッシング・モードを説明します。

LD命令は，

```
LD　デスティネーション，ソース
```

と表わされます。これはソースをデスティネーションへロードすることを意味します。BASIC風に

```
A＝B
```

とあれば，Aがデスティネーション，Bがソースとなるわけです。

1 レジスタ・アドレッシング

これはレジスタを指定するアドレッシングです。

```
例    LD  A, B
```

BASICの〝A＝B″に相当します。

2 イミディエイト・アドレッシング

直接1バイトの数値をソースとするものです。

```
例    LD  A, 50H
```

これはBASICの〝A＝&H50″に相当します。

3 エクステンド・アドレッシング

メモリのアドレスを指定するアドレッシングです。アドレスはカッコ(　)で囲んだものを記述します。

```
例    LD  A,  (1234H)
```

これは，1234H番地の内容をAに入れるものです。**図2-4**のようにメモリに書き込まれていれば，Aレジスタには23Hが入ります。また，逆にAレジスタの値が38Hのとき，

```
LD (1234H), A
```

を実行すると1234H番地の内容は38Hになります。

図2-4 エクステンド・アドレッシングの例

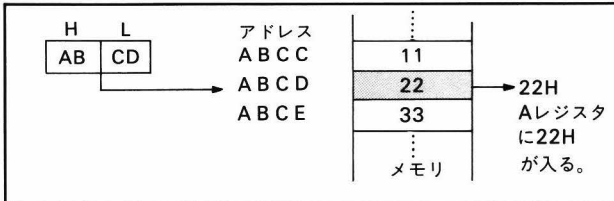
アドレス	メモリ
	⋮
1233	CD
1234	23
1235	58
	⋮

4 レジスタインダイレクト・アドレッシング

16ビット・レジスタでメモリのアドレスを指定するアドレッシングです。16ビット・レジスタの名をカッコで囲んだものを記述します。

例　　LD　A, (HL)

図2-5に例を示します。

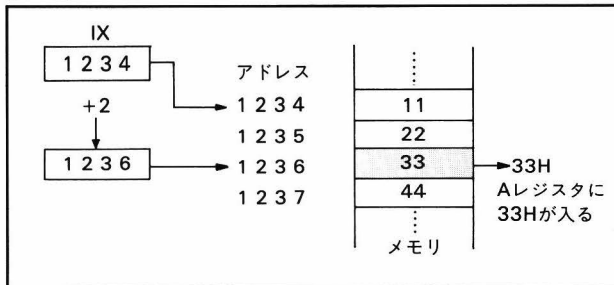
図2-5

5 インデックス・アドレッシング

インデックス・レジスタ（IX, IY）を使ってアドレスを指定します。IX, IYを使うと自身の内容を変えずに，その値を中心とする256バイトのメモリのひとつを指定できます。

例　　LD　A, (IX＋2)

図2-6に例を示します。ここで＋2をディスプレイスメント(変位）と呼びます。ディスプレイスメントは1バイトの2の補数の値なので，IXレジスタでいうと，(IX－128)から(IX＋127)の範囲を指定できます。

これまでの説明は，1バイトのアドレッシングでしたが，次に2バイトのアドレッシングについて説明します。

図2-6

6レジスタ・アドレッシング（2バイト）

16ビットのレジスタどうしのLD命令は非常に少なく，SP（スタック・ポインタ）をデスティネーションとしてHL，IX，IYのみロードすることができます。

```
例　　LD　SP，IY
```

DEレジスタの内容をBCレジスタに入れたい場合などは，他の命令を使わなければなりません。

7イミディエイト・アドレッシング（2バイト）

2の2バイト版です。BC，DE，HL，SP，IX，IYの16ビット・レジスタに直接2バイトの数値を入れるものです。

```
例　　LD　BC，1234H
```

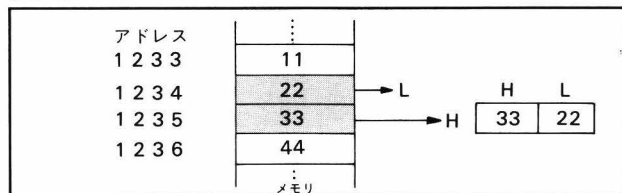
8エクステンド・アドレッシング（2バイト）

3の2バイト版です。使えるレジスタは7と同じです。

```
例1　　LD　HL，（1234H）
例2　　LD（1234H），IX
```

図2-7に例を示します。ここで注意してほしいのは，レジスタの下位バイト（図ではLレジスタ）が実際に指定したアドレスに対応し，上位バイト（図ではHレジスタ）が次のアドレスに対応することです。

いくつかの例を引きましたが，アドレッシング・モードの呼び方はとくに覚える必要はありません。ただし，カッコがついたときとそうでないときの動作の違いを把握しておいてください。

図2-7

2-2-2　Z80の命令セット

Z 80のニモニックは次の67種類があり，８つのグループに分けられます。

1データの転送，交換

LD，PUSH，POP
EX，EXX

2ブロック転送とブロック・サーチ

LDIR，LDDR，LDI，LDD
CPIR，CPDR，CPI，CPD

3演算

ADD，SUB，ADC，SBC，CP，INC，DEC
NEG，AND，OR，XOR，CPL，DAA

4ローテイト，シフト

RLC，RRC，RL，RR，SLA，SRA，SRL
RLD，RRD
RLCA，RRCA，RLA，RRA

5ビット操作，フラグ操作

BIT，SET，RES，SCF，CCF

6ジャンプ，コール，リターン

JP，JR，DJNZ，CALL，RST
RET，RETI，RETN

7入出力

IN，INI，INIR，IND，INDR
OUT，OUTI，OTIR，OUTD，OTDR

8CPUコントロール

NOP，HALT，DI，EI，IM

以上のように８つにわけたグループごとにさらに詳しくひとつひとつのニモニックを解説していきましょう。

データの転送・交換

1. 8ビット・ロード命令

ニモニックは〝LD〞で2つのオペランドを持ちます。ロード命令は,

LD　デスティネーション, ソース

と表わします。

第1オペランドがデスティネーション (受け側), 第2オペランドがソース (送り側) で, 転送はソースからデスティネーションに向かって行われます。つまり, オペランドの右側から左側に向かってロードが行われるわけです。

表2-1は8ビット・ロード命令の一覧表です。表中の

表2-1 8ビット・ロード命令

		レジスタ									メモリ						数
	ソース デスティネーション	I	R	A	B	C	D	E	H	L	(HL)	(BC)	(DE)	(IX+d)	(IY+d)	(nn)	n
レジスタ	A	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	B			○	○	○	○	○	○	○	○			○	○		○
	C			○	○	○	○	○	○	○	○			○	○		○
	D			○	○	○	○	○	○	○	○			○	○		○
	E			○	○	○	○	○	○	○	○			○	○		○
	H			○	○	○	○	○	○	○	○			○	○		○
	L			○	○	○	○	○	○	○	○			○	○		○
	I			○													
	R			○													
メモリ	(HL)			○	○	○	○	○	○	○							○
	(BC)			○													
	(DE)			○													
	(IX+d)			○	○	○	○	○	○	○							○
	(IY+d)			○	○	○	○	○	○	○							○
	(nn)			○													

●dはディスプレイスメント
●nは1バイトの値
●nnは2バイトの値

○印が存在する命令で，空欄はそれに対応する命令がないことを表わします。たとえば，〝LD　A, (DE)〟はあっても〝LD　B, (DE)〟という命令はありません。表をよく見ると，Aレジスタは万能ですが，B，C，D，E，H，Lの8ビット・レジスタには使えない命令があることがわかると思います。ペア・レジスタのなかではHLが最も機能が高くなっています。

8ビット・ロード命令は〝LD　A，I〟〝LD　A，R〟の2つを除きフラグは変化しません❺。

2. 16ビット・ロード命令

ニモニックは〝LD〟で8ビット・ロード命令と同じです。オペランドによって8ビットか16ビットかを区別します。**表2-2**は16ビット・ロード命令の一覧表です。表中の○印が存在する命令ですが，8ビット・ロード命令に比べてアドレッシング・モードが少ないこと，レジスタどうしのロード命令が少ないことがわかるでしょう。

BCレジスタの内容をDEレジスタに入れたい場合，〝LD　DE，BC〟という命令はないので，

表2-2 16ビット・ロード命令

	デスティネーション＼ソース	レジスタ BC	DE	HL	SP	IX	IY	nn	(nn)
レジスタ	BC							○	○
	DE							○	○
	HL							○	○
	SP			○		○	○	○	○
	IX							○	○
	IY							○	○
	(nn)	○	○	○	○	○	○		

```
LD  D, B
LD  E, C
```

というように，８ビット・ロード命令を使います。

16ビット・ロード命令ではフラグは変化しません。

3. PUSH，POP命令

PUSH命令には次の６個があります。

●PUSH　AF

●PUSH　BC

●PUSH　DE

●PUSH　HL

●PUSH　IX

●PUSH　IY

フラグは変化しません。

POP命令はPUSH命令と同様に次の６個があります。

●POP　AF

●POP　BC

●POP　DE

●POP　HL

●POP　IX

●POP　IY

PUSH，POP命令の動作例を**図2-8，2-9**に示します。スタックはメモリ上に設けられたエリアですから，PUSH命令はレジスタの内容をメモリに書き込むこと，POP命令はメモリからレジスタに読み込むことと同等です。誤解しやすいので念を押すと，PUSH命令を実行してもPUSHしたレジスタの内容が変わるわけではありません。

PUSH，POPはレジスタの内容を一時的に保存するために使われます。また，前に〝LD　DE，BC〟という存在しない命令を８ビット・ロード命令を使って代用しましたが，これを

```
PUSH   BC
POP    DE
```

としても同じ結果が得られます。

フラグは〝POP　AF〞を実行したとき，Fレジスタに POPした内容が入るので変化する場合があります。ほかは，フラグの変化はありません。

図2-8 PUSH命令の動作例

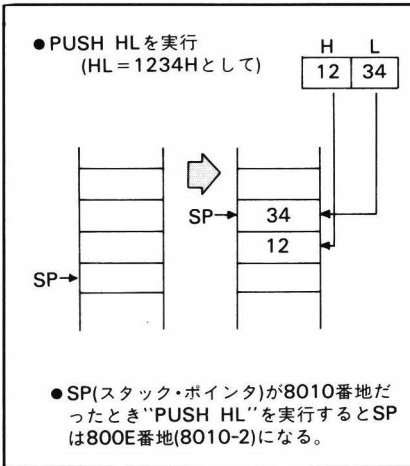
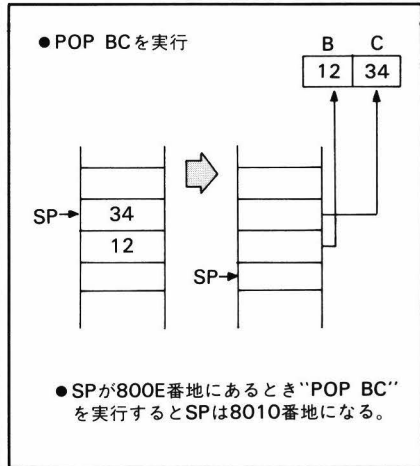



図2-9 POP命令の動作例

4. データの交換命令

交換命令には次の6個があります。

●EX　AF，AF′

●EXX

●EX　DE，HL

●EX　(SP)，HL

●EX　(SP)，IX

●EX　(SP)，IY

●EX　AF,AF′

〝EX　AF，AF′〞はメイン・レジスタのAとFをオルタ

ネート・レジスタ（裏レジスタ）のA'とF'に交換する命令です。

●EXX

〝EXX〟はメイン・レジスタの汎用レジスタ（B，C，D，E，H，L）と裏レジスタ（B'，C'，D'，E'，H'，L'）を一度に交換する命令です。PUSH，POP命令を使ってレジスタを保存，復帰するかわりに裏レジスタを使ってレジスタの保存，復帰を行う場合などに使われます。とくにEXXは，

```
PUSH   BC
PUSH   DE
PUSH   HL
   ⋮
POP    HL
POP    DE
POP    BC
```

とするかわりに，

```
EXX
 ⋮
EXX
```

のような使われ方が多いようです。もちろん，処理内容によってどちらを使うかは異なりますが…。一見便利そうなEXX命令も多用すると非常にわかりづらいプログラムになってしまう場合があります。

●EX　DE,HL

〝EX　DE，HL〟はDEレジスタとHLレジスタの内容を交換する命令ですが，HLレジスタが16ビットのアキュムレータとして使えるため，積極的に使われます。たとえば，DEレジスタの内容にBCレジスタの内容を加えるという16ビット加算命令はありませんが，

```
EX    DE, HL
ADD   HL, BC
EX    DE, HL
```

とすると望む結果が得られます(ADDは加算命令,後述)。

●EX (SP), HL ●EX (SP), IX ●EX (SP), IY

"EX (SP), HL", "EX (SP), IX", "EX (SP), IY"は，スタック・トップに積んである2バイト（つまり一番最後にPUSHした値）とHL（またはIX, IY）の内容を交換する命令です。この命令を使った例は第4章で紹介します。

ブロック転送とブロック・サーチ

1. ブロック転送

ロード命令は1命令で1バイトか2バイトのデータをレジスタとメモリの間で転送するだけでしたが，ブロック転送命令は1命令で大量のデータを転送するものです。ブロック転送命令には次の4個があります。

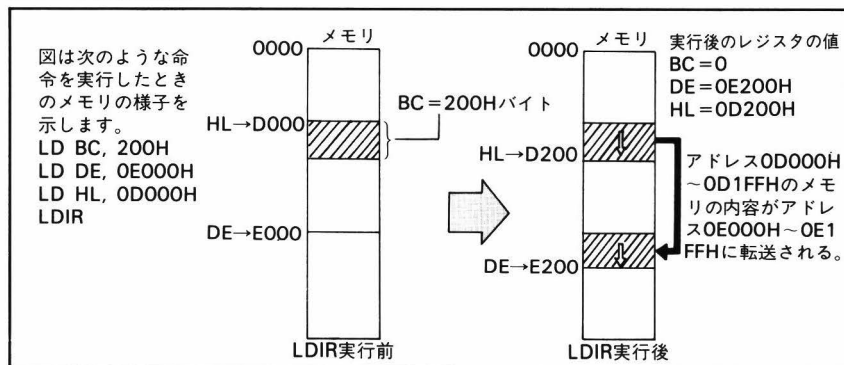
●LDIR　(LoaD, Increment and Repeat の略)
●LDDR　(LoaD, Decrement and Repeat の略)
●LDI　(LoaD and Increment の略)
●LDD　(LoaD and Decrement の略)

ブロック転送命令は，ペア・レジスタを次のように設定してから使います。

BC：転送する長さ（バイト数）
DE：転送先のアドレス
HL：転送するデータの格納アドレス

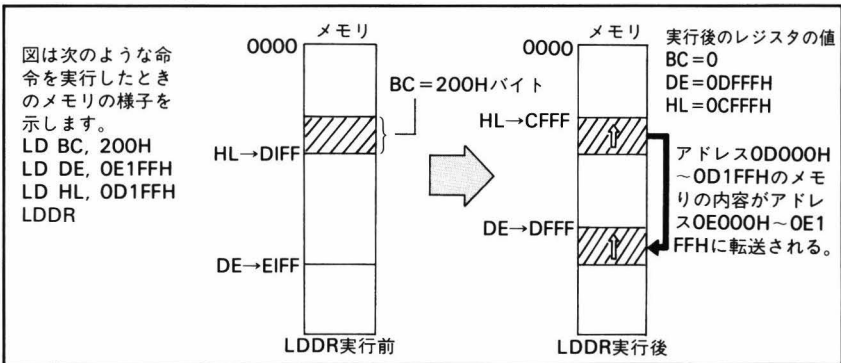
●LDIR

LDIR命令を実行するとHLが示すメモリからデータを読み出し，DEが示すメモリに転送します。1バイトのデータを転送後，HLとDEはインクリメント（＋1）さ

図2-10 LDIRの動作

れ，BCはデクリメント（－1）されます。この動作をBCの値が0になるまで繰り返します。**図2-10**に動作例を示します。

●LDDR

LDDR命令を実行するとHLが示すメモリからデータを読み出し，DEが示すメモリに転送します。1バイトのデータを転送後，HL，DE，BCがデクリメント（－1）されます。BCが0になるまで続けられます。動作例を**図2-11**に示します。

図2-11 LDDRの動作

●LDIRとLDDRの使いわけ

LDIRとLDDRは，転送する領域と転送される領域が重ならない限り，どちらの命令を使っても問題ありません。これが重なっている場合，使いわけることが必要です。例を**図2-12，2-13**に示します。

●LDI

LDI命令は，自動的に繰り返しを行わないだけで，動作はLDIR命令と同じです。つまり，HLが示すメモリからデータを読み出し，DEが示すメモリに転送します。転送後，HLとDEはインクリメント（＋1）され，BCはデクリメント（－1）するという動作を1回だけ行います。このとき，BC≠0ならばP/Vフラグがセットされます。

図2-12 LDIR命令しか使用できない場合

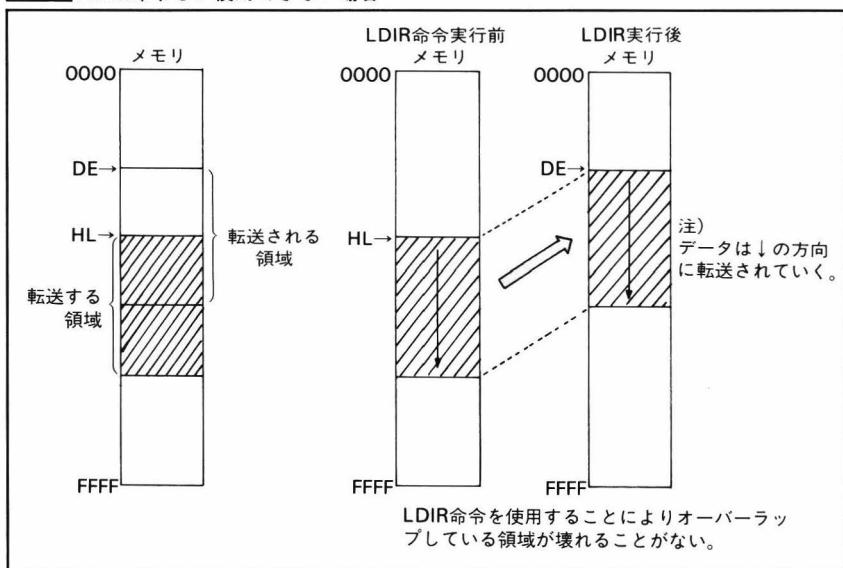
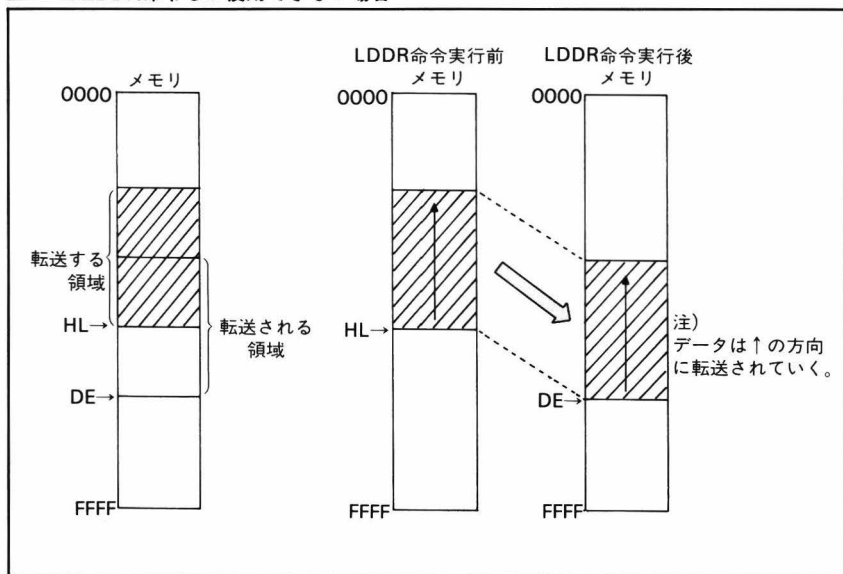



図2-13 LDDR命令しか使用できない場合

●LDD

LDD命令も，自動的に繰り返しを行わないだけで，動作はLDDR命令と同じです。つまり，HL が示すメモリからデータを読み出し，DEが示すメモリに転送します。転送後，HL，DE，BCをデクリメント（－1）するという動作を1回だけ行います。

ブロック転送命令は，LDIRとLDI，LDDRとLDD の動作，およびLDIRとLDDR，LDIとLDD の動作を対比させて覚えてください。

2. ブロック・サーチ

ブロック・サーチ命令は，メモリの中からある特定のデータをサーチするものです。ブロック・サーチ命令には次の4個があります。

●**CPIR**　(ComPare, Increment and Repeat の略)
●**CPDR**　(ComPare, Decrement and Repeat の略)
●**CPI**　(ComPare and Increment の略)
●**CPD**　(ComPare and Decrement の略)

ブロック・サーチ命令は，レジスタを次のように設定してから使います。

BC：サーチする長さ（バイト数）
HL：サーチするメモリの格納アドレス
A　：サーチするデータ

●CPIR

CPIR命令を実行すると，AレジスタとHLが示すメモリとの内容を比較し，一致していたらZフラグをセットします。次にHLはインクリメントされ，BCはデクリメントされます。BCの値が0になるか，Zフラグがセットされる（一致する）までこの動作を繰り返します。CPIR

命令を実行後，一致するデータがあればZフラグがセットされ，HLは一致したデータがあるアドレスの次のアドレスを示しています。動作例を**図2-14**に示します。

図2-14 CPIR命令の動作例

①一致するデータがある場合。

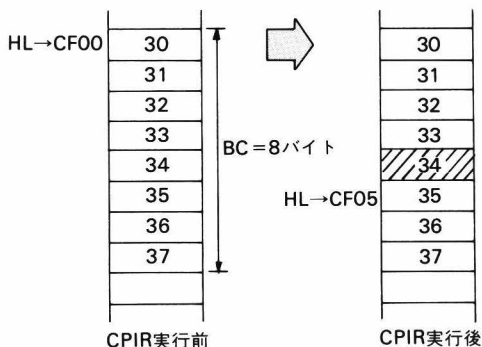
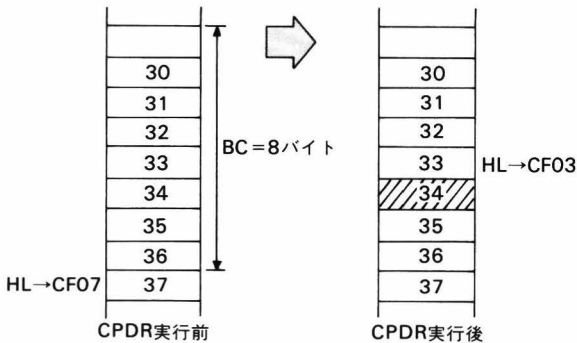
実行前のレジスタの値
A＝34H　BC＝8
HL＝0CF00H

実行後のレジスタ，フラグの値
A＝34H　BC＝3
HL＝0CF05H
Zフラグ＝1，P/Vフラグ＝1

HL→CF00
30
31
32
33
34
35
36
37

BC＝8バイト

30
31
32
33
34
HL→CF05
35
36
37

CPIR実行前　　CPIR実行後

②一致するデータがない場合。

実行前のレジスタの値
A＝34H　BC＝8
HL＝0CF80H

実行後のレジスタ，フラグの値
A＝34H　BC＝0
HL＝0CF88H
Zフラグ＝0，P/Vフラグ＝0

HL→CF80
40
41
42
43
44
45
46
47

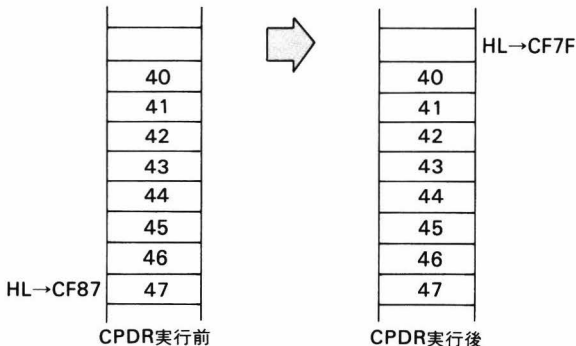
BC＝8バイト

40
41
42
43
44
45
46
47
HL→CF88

CPIR実行前　　CPIR実行後

●CPDR

CPDR命令は，比較後のHL がインクリメントではなくデクリメントされるだけで，ほかは CPIRと動作は同じです。動作例を**図2-15**に示します。

図2-15 CPDR命令の動作例

●CPI

CPI命令は繰り返しをしない以外はCPIR命令と同じ動作をします。つまり，アキュムレータ(A)とHL が示すメモリの内容とを比較し，一致していたらZフラグをセットします。次にHLはインクリメントされBCはデクリメントされます。この動作を1回だけ行います。このときBC≠0ならP/Vフラグがセットされます。

●CPD

CPD命令は，比較の後のHLのインクリメントがデクリメントになるだけで，CPI命令と同じ動作をします。

演算命令

1. 8ビット演算命令

8ビット演算命令には次の13個があります。

●**ADD**
●**ADC**　(ADd with Carry)
●**SUB**　(SUBtract)
●**SBC**　(SuBtract with Carry)
●**AND**
●**XOR**　(eXclusive OR)
●**OR**
●**CP**　(ComPare)
●**INC**　(INCrement)
●**DEC**　(DECrement)
●**DAA**　(Decimal Adjust Accumulator)
●**CPL**　(ComPLement accumulator)
●**NEG**　(NEGate accumulator)

演算命令は，オペランドがあったりなかったり，またフラグの変化が重要になったりして少々やっかいなものです。

●ADD

書式は，

```
ADD     A, opr
```

で，Aレジスタの内容とopr（oprはオペランドの略）の内容を加え，その結果をAレジスタに入れます。BASIC風に書くと，

```
A=A+opr
```

となります。oprは，A，B，C，D，E，H，L，(HL)，(IX+d)，(IY+d)，nの11種類です。たとえば，

```
ADD     A, A
```

はAレジスタの内容を2倍にすることと同じです。

フラグの変化は,

CY	：結果がFFHを超えるとセット
Z	：結果としてAレジスタに0が残るとセット
H	：ビット3からの桁あがりが入る
N	：リセットされる
P/V	：結果が2の補数の範囲（10進数で−128〜+127）を超えるとセット
S	：結果が2の補数の値として負(80H〜FFH)になったときセット

となります。

●ADC

書式は,

```
ADC     A, opr
```

で，Aレジスタとoprの内容を加え，さらにキャリーフラグの内容も加えます。結果はAレジスタに残ります。oprはADD命令と同じで，さらにフラグの変化も同じです。キャリーフラグは，0か1のどちらかですから，もしキャリーフラグが0ならADD命令と結果が同じになります。

●SUB

書式は,

```
SUB     opr
```

で，Aレジスタの内容からoprの内容を引き，結果はAレジスタに入ります。oprはADD命令と同じです。オペランドにoprだけ書くことに注意してください(Aを書く必要がない)。これは，16ビットの演算命令にSUBという命令がないためです。

フラグの変化は,

CY ：演算する前にA＜opr(符号なし整数として)
　　　のときセット
N ：セット
P/V：ADD命令と同じ
H ：ビット3のボローが発生すればセット
Z ：結果が0 (A＝opr) ならばセット
S ：ADD命令と同じ

●SBC

書式は,

```
SBC     A, opr
```

で, SUB命令と同様にAレジスタの内容からopr の内容を引き, さらにキャリーフラグの内容も引きます。この命令は, SUB命令と違ってオペランドにAを省略することはできません。最初のうちは間違えやすいので注意してください。フラグの変化はSUB命令と同じです。

●AND　●OR　●XOR

書式は,

```
AND     opr
OR      opr
XOR     opr
```

でAレジスタの内容とoprの内容を論理演算を行い,結果をAレジスタに入れます。論理演算については第1章を参照してください。oprはADD命令と同じで, オペランドにAを書く必要はありません。

フラグの変化は,

CY ：リセット
N ：リセット
P/V：演算後, Aレジスタの1になっているビット

	が偶数個（Even）であればセット
H	：AND命令ではセット，OR/XORではリセット
Z	：結果がゼロならセット
S	：結果が2の補数の値として負ならばセット

となります。

AND，OR，XOR命令は，もちろん論理演算に使われますが，Aレジスタをオペランド（Aレジスタどうしで論理演算を行う）にすることで次のような操作ができます。

1AND AまたはOR A

Aレジスタの内容を変えずにキャリーフラグのリセットができます。また，A＝00Hの場合，ゼロ・フラグがセットされ，A＝80H～FFHの場合，Sフラグがセットされる，というようにAレジスタの状態を調べることができます。

2XOR A

Aレジスタを0にします。これは，〝LD A，0〃とフラグの変化を除けば同じことです。またゼロ・フラグをセットする場合にも使われます。

●CP

書式は，

```
CP  opr
```

で，Aレジスタの内容からoprを減算しますが，結果は残りません。つまり，Aレジスタの内容は変化しません。oprはADD命令と同じ11種類で，フラグの変化はSUB命令と同じです。

CP命令は単独で使ってもあまり意味がありませんが，後述する条件分岐命令と組み合わされて使われます。

●INC

書式は，

```
INC	opr
```

で，oprの内容に1を加える命令です。oprは，A，B，C，D，E，H，L，(HL)，(IX+d)，(IY+d)の10種類です。

フラグの変化は，

```
CY ：変化しない
N  ：リセット
P/V：結果が80Hになったときセット
H  ：ビット3からのキャリーが入る
Z  ：結果が00Hになったときにセット
S  ：結果が2の補数の値として負ならばセット
```

となります。

INC命令は，ADD命令とは異なり，キャリーフラグを変化させることはありません。たとえば，A=FFHのとき，〝ADD　A，1〞を実行するとA=00H，CY=1となりますが，〝INC　A〞を実行するとA=00Hにはなってもキャリーフラグはもとのままで変化しません。

●DEC

書式は，

```
DEC	opr
```

でINC命令とは逆にoprの内容から1を引く命令です。

フラグの変化は，

```
CY ：変化しない
N  ：セット
P/V：結果が7FHになったときにセット
H  ：ビット3からのボローが入る
Z  ：}
S  ：} INC命令と同じ
```

となります。

●DAA

書式は,

```
DAA
```

でオペランドを持ちません。DAA命令は,加算命令や減算命令と組み合わせ使うことによりBCDでの10進演算ができます。加減算の命令は2進数として演算しているため,演算前の値がBCDであっても演算後の値はBCD ではなくなってしまいます。それを補正しBCDにするのがDAA命令です。

フラグの変化は,

CY	：上位4ビットを10進数に変換したとき，キャリー／ボローが発生すればセット
N	：変化しない
P/V	：命令実行後，Aレジスタの1になっているビットが偶数であればセット
H	：下位4ビットを10進数に変換したときキャリー／ボローが発生すればセット
Z	：Aレジスタが00Hになればセット
S	：Aレジスタのビット7と同じになる

となります。

●CPL

書式は,

```
CPL
```

で，オペランドを持ちません。この命令はAレジスタの全ビットを反転させるものです（1の補数をとる)。つまり，論理演算のNOTに相当します。

N，Hフラグがセットされる他は変化しません。

●NEG

書式は,

```
NEG
```

で，オペランドを持ちません。この命令は，Aレジスタに入っている値の2の補数をとります。結果はAレジスタに入ります。

フラグの変化は，

CY　：実行前にAレジスタの値が00Hならリセット，ほかの場合はセット
P/V：結果が80Hならばセット
Z　：結果が0ならばセット
S　：結果が2の補数として負ならばセット

となります。実行前にAレジスタの値が80H（－128）のときのみ，実行後も80Hになってしまいます（このときP/Vフラグがセットされる)。

2. 16ビット演算命令

16ビット演算命令は8ビット演算命令に比べて，数が少なく，オペランドも多くありません。16ビット演算命令を**表2-3**に示します。表を見るとわかりますが，ありそうでない命令（ADD　IX，HLなど）があるので注意してください。

表2-3 16ビット演算命令

命令＼オペランド	BC	DE	HL	SP	IX	IY
ADD HL,opr	○	○	○	○		
ADD IX,opr	○	○		○	○	
ADD IY,opr	○	○		○		○
ADC HL,opr	○	○	○	○		
SBC HL,opr	○	○	○	○		
INC opr	○	○	○	○	○	○
DEC opr	○	○	○	○	○	○

●ADD

書式は，

```
ADD    HL, opr
ADD    IX, opr
ADD    IY, opr
```

で，HLレジスタまたはインデックス・レジスタ（IX，IY）の内容に16ビット・レジスタの内容を加え，結果をHLレジスタまたはインデックス・レジスタに格納します。oprは**表2-3**のとおりです。

フラグの変化は，

CY：	結果が16ビットを超えたとき（FFFFHを超えたとき）セット
N ：	リセット
H ：	不定
S，Z，P/V：	どれも変化しない

となります。

●ADC

書式は，

```
ADC    HL, opr
```

でHLレジスタの内容にoprの内容を加え，さらにキャリーフラグの内容を加えます。

フラグの変化は，

CY ：	ADD命令と同じ
N ：	ADD命令と同じ
H ：	ADD命令と同じ
P/V：	結果が2の補数の範囲（－32768～＋32767）を超えたときにセット
Z ：	実行後，HLの内容が0ならばセット

S　：結果が2の補数の値として負（8000H～FFFFH）ならばセット

となり，ADD命令とフラグの変化が少々違います。

●SBC

書式は，

```
SBC     HL，opr
```

で，HLレジスタからoprの内容を引き，さらにキャリーフラグの内容も引きます。

フラグの変化は，

CY：符号なし整数としてみたとき，引く数の方が大きければ（演算前にHL<opr+CYならば）セット
N　：セット
P/V，H，Z，S：ADC命令と同じ

となります。16ビット演算命令には，8ビット演算命令のSUB命令に相当する命令はありません。したがって，SUB命令と同様の結果を得たければ，演算前にキャリーフラグをリセットする必要があります。たとえば，

```
SUB     HL，DE
```

という存在しない命令を代用するには，前に説明した論理演算命令を使って，

```
AND     A     または     OR     A
SBC     HL，DE
```

とします。

●INC，DEC

書式は，

```
INC     opr
```

```
DEC     opr
```

で，opr の内容をインクリメントまたはデクリメントします。

この命令はフラグを変化させません。とくに，8ビットのINC/DEC命令ではゼロ・フラグが変化するのに対し，16ビットのINC/DEC命令では変化しないことに注意してください。

ローテイト，シフト

1. ローテイト命令

ローテイト命令は，ビット単位で左右に回転するものです。第1章の1-6-3も参照してください。

ローテイトに関する命令は，

- **RLCA**　(Rotate Left Circular Accumulator)
- **RRCA**　(Rotate Right Circular Accumulator)
- **RLA**　(Rotate Left Accumulator)
- **RRA**　(Rotate Right Accumulator)
- **RLC**　(Rotate Left Circular)
- **RRC**　(Rotate Right Circular)
- **RL**　(Rotate Left)
- **RR**　(Rotate Right)
- **RLD**　(Rotate Left Decimal)
- **RRD**　(Rotate Right Decimal)

の10種類です。

●RLC　●RRC　●RL　●RR

書式は，

```
RLC    opr
RRC    opr
RL     opr
RR     opr
```

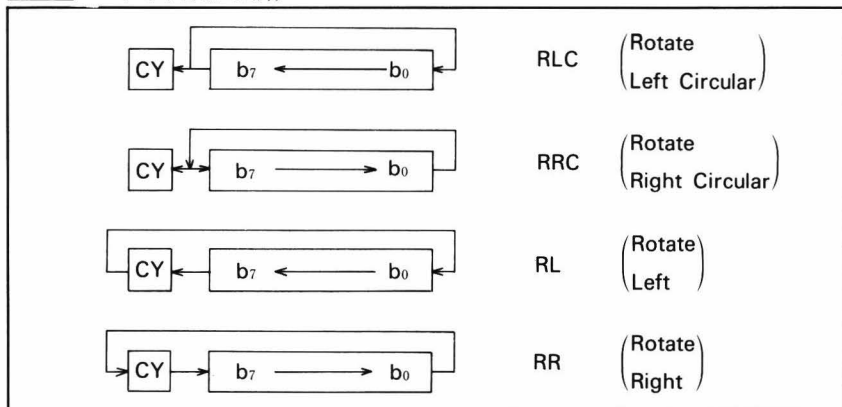
で，oprはA, B, C, D, E, H, L，(HL)，(IX+d)，(IY+d）の10種類で，oprの内容をローテイトするものです。それぞれの動作は**図2-16**を見てください。

フラグの変化は，

CY　：実行前のビット7またはビット0が入る
Z　：結果が0ならばセット

P/V：結果のビットが1になっている個数が偶数個であればセット
S　：結果が負ならばセット
H，N：リセット

となります。

図2-16 ローテイト部分の動作

●RLCA　●RRCA　●RLA　●RRA

書式は，

```
RLCA
RRCA
RLA
RRA
```

でオペランドを持ちません。Aレジスタに対してローテイトを実行する命令です。つまり，〝RLC　A〞と書いても〝RLCA〞と書いてもAレジスタに残る結果は同じです。しかし，CY，H，Nフラグの変化は同じですが，S，Z，P/Vフラグは変化しません。

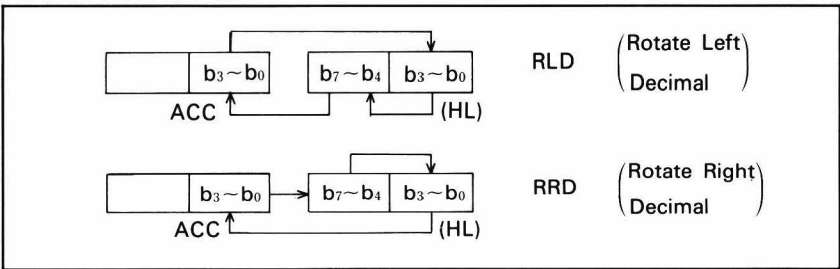
●RLD　●RRD

書式は,

```
RLD
RRD
```

で，オペランドを持ちません。これは,HLレジスタが示すメモリの内容とAレジスタの下位4ビットを左右に4ビット回転させる命令です(図2-17)。

この命令は，メモリ上に連続して記憶されているBCD表現の数値を10進数の桁単位でシフトするときに使われます。

図2-17 RLD，RRDの動作

2. シフト命令

シフトに関する命令は,

●SLA (Shift Left Arithmetic)

●SRA (Shift Right Arithmetic)

●SRL (Shift Right Logical)

の3種類です。

●SLA　●SRA　●SRL

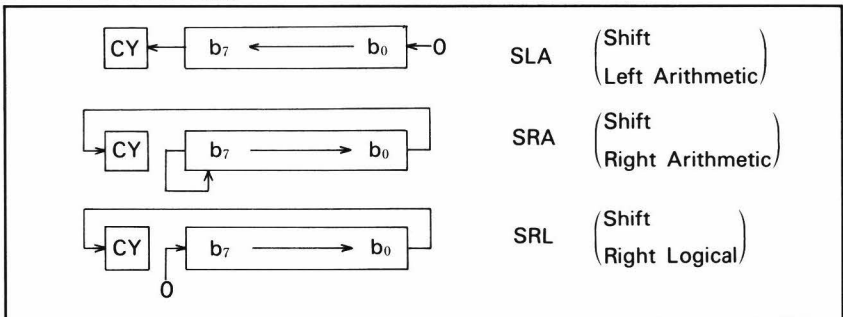
書式は,

```
SLA    opr
SRA    opr
```

SRL　　opr

で，oprはRLC命令などと同じく10種類です。これらの命令の動作を**図2-18**に示します。この図と第1章の**図1-9～1-12**と比較してみてください。SLAは算術左シフト命令と訳されていますが，実際の動作は論理左シフトです。**図1-12**に相当する本来の意味での算術左シフト命令はありません。また，論理左シフトという名前の命令もありません。混乱しやすいので注意してください。

実は，**図1-12**の算術左シフトに相当する命令は〝究極の8ビットCPU″と呼ばれる6809（FMシリーズに使われている）にもありません。少々動作が複雑なので実現できなかったものと思われます。

図2-18 シフト命令の動作

ビット操作，フラグ操作

1. ビット操作

ビット操作命令は，

●BIT　(test BIT)

●RES　(RESet bit)

●SET　(SET bit)

の3種類です。

●BIT

書式は，

```
BIT    opr 1 , opr 2
```

で，2つのオペランドを持っています。opr 1 は0～7の数字で，何ビット目かを指定するものです。opr 2 は，レジスタまたはメモリで，A, B, C, D, E, H, L, (HL), (IX+d), (IY+d) の10種類があります。

この命令は，opr 2 のopr 1 で指定されたビットが0ならばZフラグをセット，1ならばZフラグをリセットするものです。たとえば，Bレジスタの5ビット目が0か1かを調べるためには，

```
BIT  5, B
```

として，Zフラグを調べれば良いことになります。

●SET　●RES

書式は，

```
SET    opr 1 , opr 2
RES    opr 1 , opr 2
```

で，opr 1，opr 2 はBIT命令と同じです。この命令は，opr 2 のopr 1 で指定されたビットをセットまたはリセットします。

2. フラグ操作命令

フラグを直接操作する命令は,

●CCF (Complement Carry Flag)

●SCF (Set Carry Flag)

の2種類で,キャリーフラグに対してのみ操作が可能です。

●CCF ●SCF

書式は,

```
CCF
SCF
```

でオペランドを持ちません。CCF命令はキャリーフラグの反転,SCF命令はキャリーフラグをセットする命令です。前に論理演算の〝AND A〟または〝OR A〟を使ってキャリーフラグをリセットしましたが,

```
SCF
CCF
```

としてもキャリーフラグをリセットできます。

ジャンプ，コール，リターン

1. ジャンプ命令

ジャンプ命令はプログラムの流れを変えるための命令，つまりPC（プログラム・カウンタ）を操作するための命令です。

●JP

書式は，

```
JP    nn
JP    cond, nn
JP    (HL)
JP    (IX)
JP    (IY)
```

です。ここでnnは2バイトでアドレスを表わし，condは条件名です。JP命令には，無条件ジャンプと条件によってジャンプする条件ジャンプがあり，条件ジャンプは条件が合わなければ，ジャンプせずに次の命令を実行します。条件には，**表2-4**に示すようなものがあります。

例） AレジスタとBレジスタを比較し，もし等しければ8000番地へジャンプし，Bレジスタの内容の方が大きけ

表2-4 条件ジャンプの条件名

条　件　名	成　立　条　件
C (Carry)	CYフラグ＝1
NC (Non Carry)	CYフラグ＝0
Z (Zero)	Zフラグ＝1
NZ (Non Zero)	Zフラグ＝0
PE (Parity Even)	P/Vフラグ＝1（パリティ偶数またはオーバーフローあり）
PO (Parity Odd)	P/Vフラグ＝0（パリティ奇数またはオーバーフローなし）
M (Minus)	Sフラグ＝1（2の補数で負）
P (Plus)	Sフラグ＝0（2の補数で正またはゼロ）

れば8100番地へジャンプし，どちらでもなければ8200番地へジャンプするというプログラムは，

```
CP    B
JP    Z, 8000H      (A=Bの場合)
JP    C, 8100H      (A<Bの場合)
JP    8200H
```

となります。

JP (HL), JP (IX), JP (IY) は，それぞれHL, IX, IY レジスタが示すアドレスへ無条件にジャンプする命令です。

●JR　●DJNZ

書式は，

```
JR    d
JR    cond, d
DJNZ  d
```

で，dは2の補数の1バイト，condは条件名です。これらはジャンプ命令に対して，相対ジャンプ命令と呼ばれ，dはディスプレイスメントでJR命令が置かれているアドレスを基準として－126～＋129の範囲の値です。したがって，JP命令のように全アドレスへジャンプできるわけではありません。

条件付きのJR命令の条件はJP命令より少なく，**表2-5**に示すように4種類です。

DJNZ命令は，条件付きのJR命令で，まずBレジスタ

表2-5 条件付き相対ジャンプ命令の条件名

条件名	成立条件
C (Carry)	CY=1
NC (Non Carry)	CY=0
Z (Zero)	Z=1
NZ (Non Zero)	Z=0

から１を引き結果が０でなければジャンプするという動作をします。ですから，この命令は，

```
DEC     B
JR      NZ, d
```

と等価です。ただし，DJNZ 命令ではフラグは変化しません。

2. コール，リターン命令

コール，リターン命令は，BASICの〝GOSUB〞，〝RETURN〞と同様の動作をする命令です。

●CALL

書式は，

```
CALL    nn
CALL    cond, nn
```

で，nnはアドレスを表わす２バイトの数値，condは条件名です。条件名は，JP命令と同じです。条件が合わないときは何もせず次に進みます。

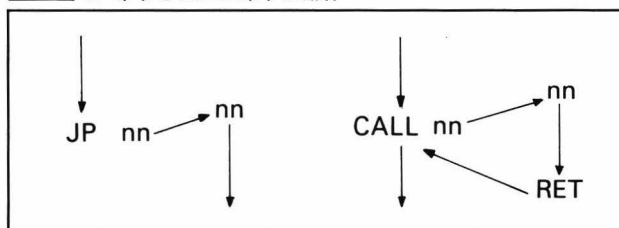
●RET

書式は，

```
RET
RET     cond
```

で，サブルーチンから戻る命令です。CALL命令で呼ばれたサブルーチンはRET命令で戻らなければなりません。

JP命令とCALL命令によるプログラムの流れを**図2-19**に示します。CALL命令を実行すると，スタックにはCALL命令の次のアドレスがPUSHされます。RET命令を実行するとスタックの先頭アドレスから戻りアドレスをPOPします。

図2-19 JP命令とCALL命令の動作

条件付きRET命令の条件は，JP命令と同じです。

●RST

RSTはReSTartの略で，書式は，

```
RST  n
```

です。リスタート命令は，コール命令と同様の動作をしますが，コールできるアドレスが決まっています。nの値とコールするアドレスは**表2-6**のようになっています。

表2-6 RSTの命令

n	コールするアドレス
00H	0000H
08H	0008H
10H	0010H
18H	0018H
20H	0020H
28H	0028H
30H	0030H
38H	0038H

入出力命令

1. IN, OUT命令

入出力命令は，I/Oポートを介してCPUと外部機器とのデータの受け渡しを行う命令です。

●IN

書式は，

```
IN    A, (n)
IN    reg, (C)
```

で，nはI/Oポート番号，regは8ビット・レジスタ（A, B, C, D, E, H, L）です。前者は，nで指定されたI/OポートからAレジスタに1バイトのデータを入力する命令で，後者はCレジスタで指定されたI/Oポートからregに1バイトのデータを入力する命令です。〝IN A,(n)〟ではフラグは変化しませんが，〝IN reg, (C)〟を実行したときのフラグの変化は，

CY	：変化しない
N	：リセットされる
P/V	：入力データのうち1になっているビットが偶数個あればセット
H	：リセット
Z	：入力データが0ならセット
S	：入力データのビット7が1ならばセット

となっています。

●OUT命令

書式は，

```
OUT    (n), A
OUT    (C), reg
```

で，nはI/Oポート番号，regは8ビット・レジスタです。

IN命令とは逆にI/Oポートにデータを出力する命令です。この2つの命令はフラグに影響を与えません。

2. ブロック入出力命令

ブロック入出力命令は，レジスタを次のように設定してから実行します。

B	：バイト数
C	：I/Oポートのアドレス
HL	：入力の場合は入力データの転送先アドレス，出力の場合は出力データの格納アドレス

なお，ブロック入出力命令にオペランドはありません。

●INIR (INput Increment and Repeat)

INIR命令は，レジスタCが示す入力ポートからデータを入力し，ペア・レジスタHL が示すメモリへ転送します。次にペア・レジスタHLはインクリメントされ，レジスタBはデクリメントされます。B=0になるまでこの動作を繰り返します。

●INDR (INput, Decrement and Repeat)

INDR命令は，レジスタCが示す入力ポートからデータを入力し，ペア・レジスタHLが示すメモリへ転送します。次にペア・レジスタHLとレジスタBをデクリメントします。B= 0 になるまでこの動作を繰り返します。

●INI (INput and Increment)

INI命令は，レジスタCが示す入力ポートからデータを入力し，ペア・レジスタHLが示すメモリへ転送します。次にペア・レジスタHLをインクリメントし，レジスタBをデクリメントします。なお，このときB = 0 になればZフラグがセットされます。

●IND (Increment and Decrement)

IND命令は，レジスタCが示す入力ポートからデータ

を入力し，ペア・レジスタHLが示すメモリへ転送します。次にペア・レジスタHLとレジスタBをデクリメントします。なお，このときB＝0になればZフラグがセットされます。

●**OTIR** (OuTput, Increment and Repeat)

OTIR命令は，ペア･レジスタHLが示すメモリの内容をレジスタCが示すポートへ出力します。次にペア・レジスタHLはインクリメントされ，レジスタBはデクリメントされます。B＝0になるまでこの動作を繰り返します。

●**OTDR** (OuTput, Decrement and Repeat)

OTDR命令は，ペア・レジスタHLが示すメモリの内容をレジスタCが示すポートへ出力します。次にペア・レジスタHLとレジスタBをデクリメントします。B＝0になるまでこの動作を繰り返します。

●**OUTI** (OUTput and Increment)

OUTI命令は，ペア・レジスタHLが示すメモリの内容をレジスタCが示すポートへ出力します。次にペア・レジスタHLをインクリメントし，レジスタBをデクリメントします。なお，このときB＝0になればZフラグがセットされます。

●**OUTD** (OUTput and Decrement)

OUTD命令は，ペア・レジスタHLが示すメモリの内容をレジスタCが示すポートへ出力します。次にペア・レジスタHLとレジスタBをデクリメントします。なお，このときB＝0になればZフラグがセットされます。

CPUコントロール命令

CPUをコントロールする命令には，次の５種類があります。

●**NOP** (No OPeration)

NOP命令は何の動作もしない命令です。

●**HALT** (HALT)

HALT命令を実行するとCPUはこの命令が書かれているアドレスで停止します。この状態は，CPUのリセット信号が入力されるか，割り込みがかかるまで続きます。

●**DI** (Disable Interrupt)

マスカブル割り込みによる割り込みを禁止します。

●**EI** (Enable Interrupt)

マスカブル割り込みによる割り込みを可能にします。

●**IM** (Interrupt Mode)

割り込みモードを設定する命令で，モードは０，１，２の３種類があります。X１はモード２の割り込みを使っています。

用語解説

❶プログラム・カウンタ

BASICのGOTO，GOSUB，RETURNに相当する命令として，Z80にはジャンプ，コール，リターンという命令がある。プログラム・カウンタは，通常自動的にインクリメントされるが，これらの命令で強制的にプログラム・カウンタの値を変えることができる。

❷割り込み（インタラプト）

割り込みというのは，ある処理の実行中に他の処理を割り込ませて実行すること。Z80にはノンマスカブル（禁止できない）割り込み（NMI）とマスカブル（禁止できる）割り込みの2種類の割り込みがあり，そのうちマスカブル割り込みには3種類の割り込みモードがある。ノンマスカブル割り込みはソフトで禁止することはできないが，マスカブル割り込みは，〝DI〞，〝EI〞という命令があり，割り込みの禁止と解除が可能。

Z80では割り込みがハードウェアによって起こるが，通常のプログラムであれば割り込みのことを気にする必要はない。

❸ダイナミックRAM

RAMにはダイナミックRAM(DRAM)とスタティックRAM(SRAM)の2種類がある。DRAMは一定時間ごとに記憶内容を復習(これをリフレッシュと呼ぶ）させてやらないと忘れてしまう。

❹リフレッシュ・アドレス

次にどのメモリ・ブロックをリフレッシュするかを示す。

❺LD A,IとLD A,Rのフラグの変化

この2つの命令を実行したときのフラグの変化は，

CY ：変化しない
N,H：リセットされる
P/V：割り込み禁止のときリセット，割り込み許可のときセット
Z ：Aレジスタに0が残ればセット
S ：Aレジスタに2の補数で負であればセット

となっている。したがって，実行後P/Vフラグを見ると割り込みが禁止されているか許可されているか知ることができる。

第3章

エディタ・アセンブラ

3-1 概要
3-2 運用法
3-3 エディタ
3-4 アセンブラ
3-5 ローダー
3-6 モニタ
3-7 アセンブラの文法

3-1 概要

本書のエディタ・アセンブラの特徴は次のとおりです。

1エディタは，文字列のファインド(検索)，リプレイス(置き換え) やブロック単位の編集 (転送やコピー) などが可能で，ワードスター❶のようなコマンドを持つ強力なフルスクリーン・エディタです。

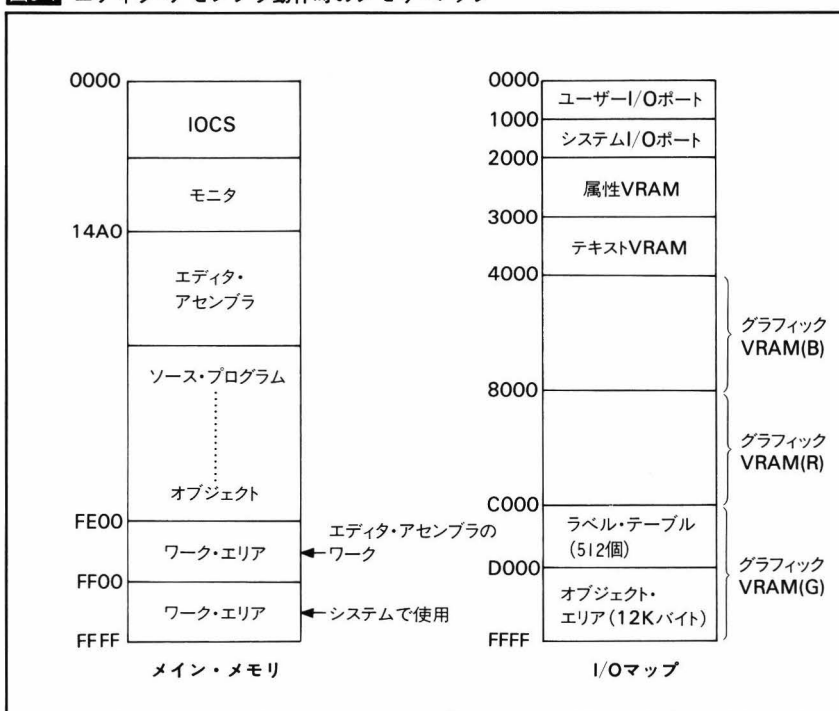
2アセンブラはザイログ・ニモニック準拠 (第2章で解説したニモニック) のアセンブラで，オンメモリ❷のため高速です。

3アセンブルしたオブジェクト (マシン語) は，グラフィックVRAMに出力し，ローダーによりメイン・メモリにロードします (このためX1ではグラフィックRAM が必要です)。

4モニタは， X1のBASICのモニタ機能にブレーク・ポイント (後述) の設定，レジスタ表示・変更の機能を付加し，デバッグには威力を発揮します。

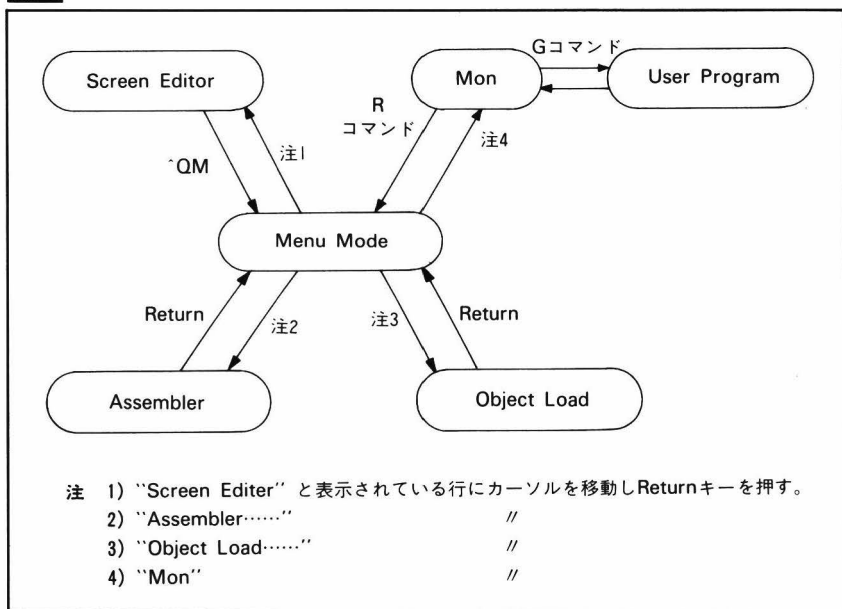
5X1の広いメモリ空間を利用し，ソース・プログラム❸のエリアは約40KBほどあります (**図3-1**)。

これらの特徴に，X1のカセット・システムの使いやすさを併わせると，マシン語プログラムの開発にはほぼ十分であると考えられます。

図3-1 エディタ・アセンブラ動作時のメモリ・マップ

3-2 運用法

エディタ・アセンブラを起動するとメイン・メニューが表示されます（この状態を〝メニューモード〟と呼びます)。メニューモードから各処理（スクリーン・エディタ，オブジェクト・ロード，モニタ）に制御を移します（図3-2)。

マシン語プログラムを開発するためには，次の順序で行います。

図3-2

1スクリーン・エディタでソース・プログラムを入力する。

2メニューモードに戻り，アセンブラに制御を移す。

3アセンブル後，エラーがあればスクリーン・エディタに制御を移し，エラー行を訂正する。エラーがなくなれば，アセンブル後，オブジェクト・ローダーでオブジェクトをVRAMからメイン・メモリにロードする。

4モニタに制御を移し実行する。ただし，暴走する可能性があるので（というより一度で動くことは，まずあり得ない），実行する前にソース・プログラムをカセットにセーブする。

3-3 エディタ

メニューモードの状態でカーソルを〝Screen Editor″と表示されている行に移動して[↵]キーを押すとエディタに入ります。このとき，すでにプログラムがあると表示されます。

画面の最上段に表示されているのは，カーソルの表示されているカラム，ラインと未使用領域（バイト数）です。さらにインサート・モードＯＮの状態を示しています。

エディタのコマンド表を**表3-1**に示します。このコマンド表は^J（[CTRL]を押しながら[J]を押すという意味）で画面に表示されます。コマンドは，

●カーソル移動

●インサート＆デリート

●ファインド＆リプレイス

●ブロック・オペレーション

●その他

に大別できます。

3-3-1 カーソル移動

カーソルを移動させるには，次のコマンドまたはキーを使います。

[1]カーソル・キー

カーソル・キーで上下に１行，左右に１文字単位で移動します。BASICを使っているときのカーソル・キーとは少々異なり，文字のないところにはカーソルを移動できません。プログラムを１行打ち込んだら[↵]キーを押しますが，[↵]のASCIIコードも１文字として扱います。

表3-1 コマンド表

●カーソル移動

^S, ←	Left Character	^D, →	Right Character
^A	Left Word	^F	Right Word
^E, ↑	Up Line	^X, ↓	Down Line
^R	File Up Screen	^C	File Down Screen
^W	File Up 1 Line	^Z	File Down 1 Line
^QR	To Top of File	^QC	To End of File
^QB	Begin Marker	^QK	End Marker
^I,HTAB	TAB Set	^H,DEL	Left Character

●インサート & デリート

^V	Insert Mode On/Off
^G	Delete Character under Cursor
^Y	Delete Line
^T	Delete to End of Line

●ファインド & リプレイス

^QF	Find
^QA	Find & Replace
^L	Find/Replace Again

●ブロック・オペレーション

^KB	Begin Block	^KK	End Block
^KC	Copy Block	^KV	Move Block
^KY	Delete Block	^KH	Hide Marker
^KS	Save Text	^KR	Read Text
^KP	Print Block	^KQ	Abandon a File

●その他

^P	Print Text
^J	Display Help File
^QM	Return to Main Menu

2カーソル・キーと同じ動作をするもの

^Eで上，^Xで下，^Dで右，^Sで左へカーソルが移動します。これらのキーは[CTRL]キーの近くにあり，左手の小指で[CTRL]キーを押しながら人差指や中指を使って片手で操作できます。

3スクロール

^Wで上，^Zで下へ1行分スクールします。また，^Rで上，^Cで下へ12行分いちどにスクロールします。

4ワード単位のカーソル移動

^Fで1語分右へ，^Aで1語分左へ移動します。

5タブ

^I，HTABキーでカーソルが8カラム移動します。これを使ってリストを見やすくできます。また，タブも1個の文字（ASCIIコードの09H）として見なしています。

6その他

ファイル（この場合，ソース・プログラムのこと）の先頭へ移動するときは，^QR（^QRは^Qを押すと画面の左上に^Qと表示されて次のコマンドを待つので，ここでRまたはrを押す。または^Q^Rと押してもよい。^QCや^KKなども同様）で，ファイルの終わりへは^QCで移動します。また，^QBはブロックの始め、^QKはブロックの終わり（ブロックはブロック・オペレーションを参照のこと）へ移動します。

3-3-2 インサート&デリート

ここで説明するコマンドは，文字や行の挿入や削除に使います。

1インサート・モードの切り換え

エディタの起動時は，インサート・モードがONになっています（最上段に反転文字で表示)。インサート・モードのON／OFFの切り換えは^Vで行います。

②1文字削除

^Gでカーソルの下1文字を消します。行の最後には，リターン・コード(ODH)が入っているので，これを^Gで消すと次の行が連結されます。

③1行削除

^Yでカーソルのある行を削除します。

④カーソル以後の削除

^Tでカーソルがある位置から行の終わりまで削除します。^Yと混同しないようにしてください。

3-3-3 ファインド&リプレイス

これはソース・プログラムの中から任意の文字列を検索（ファインド）したり，他の文字列に置き換える（リプレイス）コマンドです。

①ファインド

^QFを押すと画面の左上で〝Find？〞と聞いてくるので探したい文字列を入れ⏎キーを押します。このときの文字列の訂正はDELキーを使います（カーソル・キーは受け付けない)。

次に〝Options？　(？　For　Info)〞と聞いてきます。ここで単に⏎キーを押すとカーソルのあった位置からファイルの最後まで文字列を探し，最初に見つかった文字列にカーソルが移動します。n（nは1～99までの数字)を入れ⏎キーを押すとカーソルのあった位置から文字列を探し，n番目に見つかった文字列のところにカーソルが移動します。

②リプレイス

^QAを押すとファインドと同様に〝Find？〞と聞いてくるので置き換えたい文字列を入力します。次に〝Replace？〞と聞いてくるので置き換える文字列を入力します。さらに〝Options？（？　For Info)〞と聞いてきます。ここで，単に⏎キーを押すと文字列を探して見つか

った場合，画面の右上で〝Replace (Y/N)〞と聞いてくるのでYを押すとリプレイスが行われ，Nを押すとリプレイスは行われません（YとNは小文字でも可)。オプションは，他に？，n（nは1～99までの数字)，G（またはg)，N（またはn)の4種類あります。？を押すとオプション・インフォメーションを表示します。数字nは，n回リプレイスを繰り返し，Gはファイルの始めから終わりまでリプレイスします。Nを指定するとリプレイスするか否かは聞いてきません。これらは組み合わせて使うことができ，たとえばGNとすると全ファイルを確認なしでリプレイスします。なお，中断はSHIFT＋BREAKです。

3検索，置き換えの再実行

^Lで最後に実行した検索や置き換えを再実行します。このコマンドにより，同じ文字列を検索するときなどいちいち文字列を指定する必要がなくなります。

3-3-4 ブロック・オペレーション

ブロック・オペレーションは，ファイルの一部を移動，コピー，削除などをしたり，プログラムのセーブ，ロードに使われます。ブロックを指定するためには，ブロックの始めのマークと終わりのマークを付けます。以後，マークされたブロックをコピーしたり移動したりします。

1ブロックの始点をマーク

^KBでブロックの始点をマークします。マーカーは，〝→〞で示されます。

2ブロックの終点をマーク

^KKでブロックの終点をマークします。マーカーは，〝←〞で示されます。

3コピー

^KCでマークしたブロックをカーソルの位置から先にコピーします。

4移動

^KV でマークしたブロックをカーソルの位置から先に移動します。

5削除

^KYでマークしたブロックを削除します。

6マーカーを消す

^KHで，マーカーを消します。マーカーは，^Gや^Yで消すこともできます。

7ブロックの印字

^KPでマークしたブロックをプリンタに印字します。

8プログラムのセーブ

^KSでソース･プログラムをすべてカセットにセーブします。

9プログラムのロード

^KRでカセットからプログラムをロードします。ファイル名を指定しなければ最初のファイルをロードします。ロードされたプログラムは，プログラムの最後にアペンドされます。

10削除

^KQでプログラムをすべて削除します。危険なコマンドなので確認してきます。

3-3-5　その他

1ヘルプ

^Jで表3-1のようなコマンド表が表示されます。これによりコマンドを忘れたときは，マニュアルを見なおす必要はありません(もっとも^Jというコマンドを忘れたときは論外ですが…)。

2プログラムの印字

^Pでプログラムをすべてプリンタに印字します。

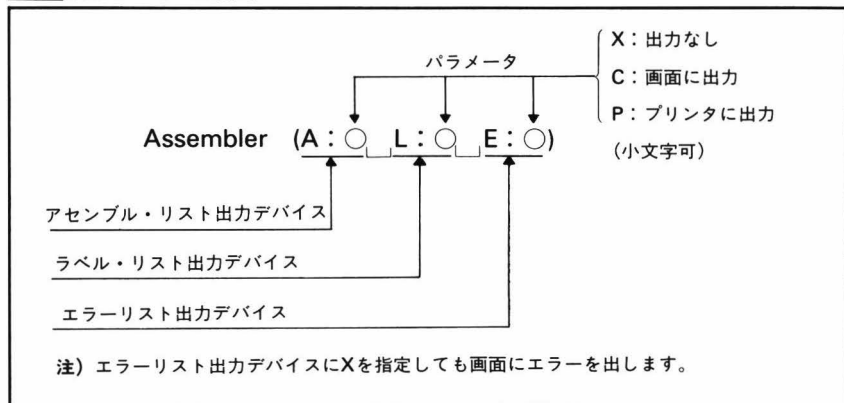
3メニューモードへ戻る

^QMでメニューモードへ戻ります。

3-4 アセンブラ

このアセンブラでは最大12Kバイトのオブジェクトを作成できます。アセンブラは，メニューモードの状態で〝Assembler……〟と表示している行にカーソルを移動し，⏎キーを押すと起動します。このとき，リスト出力デバイスを指定することで，アセンブル・リスト，ラベル・リスト，エラーリストを画面またはプリンタに出力できます。出力デバイスの指定方法は**図3-3**を見てください。

なお，画面へのリスト出力の一時停止はスペース・キー，中断はSHIFT＋BREAKで，プリンタへの出力の中断もSHIFT＋BREAKで行います

図3-3 出力デバイスの指定

3-5 ローダー

ローダーはアセンブラによってグラフィックRAMへ出力されたオブジェクトをメイン・メモリにロードするものです。ローダーはメニューモードの状態で,〝Loader……〟と表示されている行にカーソルを移動して⏎キーを押すと起動されます。

ロードできる領域は,メニューモードで表示されている〝Free Area (XXXX-FDFF)〟という範囲です。(図3-4)。Free Areaはシステム領域やソース・プログラムを壊さずにロードできる領域のことです。この範囲外にロードしようとすると,

1 "Load error"

2 "Destroy a File (Y/N)?"

という2種類のメッセージが表示されます。1はシステム領域(IOCSやアセンブラなどのプログラムがある領域)にロードしようとしたときのメッセージでロードはできません。2はソース・プログラムが存在する領域にロードしようとするときに表示されるメッセージで,Yを押すとロード,Nを押すとロードされません。当然,Yを押すとソース・プログラムが壊れるので,その前に必ずセーブしておいてください。

オブジェクトをロードする番地はソース・プログラムのORG命令で示された番地です。しかし,オフセット機能を使うとロードする番地を変更することができます。

図3-4 ローダー

オフセットは〝Offset：0000H″と表示している〝0000″のところへカーソルを持っていき指定します。この場合，ロードする番地は，

ORGで指定した番地＋オフセット値

となります。たとえば，ORG　8000Hとある場合，オフセットに1000を指定すると9000番地から，F000と指定すると7000番地からロードされます。

3-6 モニタ

モニタは，メニューモードの状態から〝Mon″と表示されている行にカーソルを移動して⏎キーを押すと起動します。モニタのコマンドはBASICのモニタとほとんど同じですが，RとGコマンドが少々異なり，新たにXコマンドが追加されました。

●Rコマンド

このコマンドを実行するとメニューモードに戻ります。

●Gコマンド

これは指定したサブルーチンをコールするコマンドですが，ブレーク・ポイントを2個まで設定することができます。Gコマンドには次の3種類があります。

1 G n

2 G n　b

3 G n　b　b'

（ここで，n，b，b'は16進4桁の数値）

1はnで示されたプログラムをサブルーチン・コールするもので，nを省略した場合，現在のプログラム・カウンタが示すアドレスからのプログラムをコールします。プログラム・カウンタの値は後述のXコマンドで見ることができます。

2はブレーク・ポイントを1個設定したもので，bで示される番地に達したときプログラムの実行がストップし，そのときのレジスタとフラグが表示されます。bの位置にある命令は実行されません。nを省略したときは1と同じです。

3はbとb'という2個のブレーク･ポイントを設定するもので，どちらかのブレーク・ポイントに達したときプログラムはストップされます。

●Xコマンド

このコマンドでレジスタ，フラグの内容を表示・変更することができます。Xコマンドには2種類あります。

1 X

2 Xレジスタ名

1はレジスタ，フラグの内容を表示するものです。フラグは2進数でも表示されます。ただし，裏レジスタの内容は表示しません。

2は指定されたレジスタの内容を変更するもので，指定できるレジスタはA, F, B, C, D, E, H, L, AF, BC, DE, HL, IX, IY, SP, PC です。

3-7　アセンブラの文法

ここでは，ソース・プログラムを作成する上で，守らなければならない規則や擬似命令，エラーメッセージなどについて解説します。

3-7-1　文の構成

ソース・プログラムの1行は次の形式で記述します。

ラベル：ニモニック␣オペランド；コメント

1ラベル

ラベルは英数字および〝?″，〝@″で構成された文字列です。ただし，先頭の文字に数字を使うことはできません。ラベルは先頭から6文字が有効で，7文字以降は無視されます。また，ラベルに含まれる英小文字は英大文字と同等に処理されます。つまり，〝LABEL″と〝Label″というラベルは同じものと見なされます。

●正しいラベルの例

LDHL

@38

?abcdefg …先頭から6文字（?abcde）がラベルと見なされる

●誤ったラベルの例

1LABEL…先頭が数字である

LA BEL …スペースが入っている

HL…………レジスタ名が使われている

LD…………ニモニックはラベルに使えない

通常ラベルの直後にはコロン（：）が必要ですが，擬似命令EQU(後述)の場合はコロンは必要ありません。

〈例〉 LABEL：LD　A，10…………LABELというラベルには現在のアドレスが割り付けられる。

LABEL　EQU　10……………LABELというラベルに定数10が割り付けられる。

2ニモニック

ニモニックについては第2章を参照してください。なお，小文字でもかまいません。

3オペランド

ニモニックによってオペランドが必要なものとないものがあります。ニモニックとオペランドの間には必ず1個以上のスペース（TABでも可）を入れてください。

●正しい例

LD　　A，10

CALL　LABEL

●誤った例

LDB，A

JPLABEL

4コメント

セミコロン（;）以降は何を記述してもかまいません。もちろん省略も可能です。しかし，コメントはプログラムを理解しやすくするために積極的につけた方が良いでしょう。

3-7-2　擬似命令

擬似命令はアセンブラに対して指示を与えるための命令です。擬似命令には次の7種があります。

●ORG

●EQU

●DEFB

●DEFM

●DEFW

●DEFS
●END

●ORG (ORiGin)

アセンブラに対して，どのアドレスからオブジェクトを出力するか指示する命令です。オペランドに式を書くこともできます。

〈例〉　ORG　4000H
　　　　ORG　START
　　　　ORG　4000H＋100H
　　　　ORG　START＋100H

●EQU (EQUate)

オペランドの値を持つラベルを定義します。したがって必ずラベルを書かなければなりません。また，EQU命令だけはラベルの直後のコロンは必要ありません（コロンを付けるとエラーと見なされます)。

〈例〉　LABEL　EQU　4020H
　　　　CR　　EQU　13
　　　　X1C　 EQU　X1＋VRAM

●DEFB (DEFine Byte)

メモリに1バイトのデータを設定します。オペランドはカンマ（，）で区切れば複数個記述することができます。

〈例〉　DEFB　10, 'A', 'B'
　　　　DEFB　BYTE…………BYTEはラベル名
　　　　DEFB　89AH…………このときオブジェクト
　　　　　　　　　　　　　　は下位2桁の9AHだけ
　　　　　　　　　　　　　　が出力される

●DEFM (DEFine Message)

メモリに文字列のデータを設定します。ただし，引用符を2つ続けて書くことで1つの引用符とします。オペ

ランドは１個だけです。

〈例〉　DEFM　'Test'

●DEFW (DEFine Word)

メモリに16ビットのデータを設定します。オペランドはカンマ（，）で区切れば複数個記述できます。メモリには上位８ビットと下位８ビットが逆に出力されます。

〈例〉　DEFW　9000H………オブジェクトは00，90と出力される

DEFW　'AB',WORD…………WORD はラベル名

●DEFS (DEFine Storage)

メモリにオペランドで示された大きさのエリアを確保します。オペランドは１個だけです。

〈例〉　DEFS 100………………100バイト分のエリアを確保する

DEFS AREA……………AREAはラベル名

●END

アセンブラに対してソース・プログラムの終了を知らせます。プログラムの途中にEND命令があれば，これ以降はアセンブルされません。

3-7-3 オペランド

オペランドにはレジスタ名や式を書きます。式は次のようなもので構成されます。

●定数

定数には数値定数と文字定数があります。数値定数は10進定数と16進定数の２種類で，16進定数は最後にＨをつけて10進定数と区別します。ただし，Ａ－Ｆで始まるときは，ラベル名と区別するため前に０をつける必要が

あります。

文字定数は，引用符（'）で囲まれた1文字または2文字のことで，値はASCIIコード表で決められます。引用符は2つ続けて書くことにより1つの引用符となります。

〈例〉　10進定数　123
　　　　　　　　456D……最後にDをつけても10進定数と見なされる
　　　16進定数　123H
　　　　　　　　0ABH
　　　　　　　　0FFFFH
　　　文字定数　'A'
　　　　　　　　'AB'
　　　　　　　　''''…………1つの引用符となる

●ラベル

式のなかでラベルを使うときは，必ず定義されたものでなければなりません。

●項と項の加算と減算

〈例〉　・LD　B，0F0H＋4
　　　　LD　C，L1＋L2－L3…L1,L2,L3はラベル

●＄（ロケーション・カウンタ）

現在アセンブル中の命令の番地を与えます。たとえば〝LD HL，＄〟という命令があり，アセンブル時にこれがD308番地とすると，オブジェクトは〝2108D3〟と出力されます。

〈例〉　＄－12
　　　　＄＋L1……L1はラベル

3-7-4　リストについて

リスト3-1はソース・リスト（エディタでつくったリスト），**リスト3-2**はアセンブル・リスト，**リスト3-3**はラベル・リストです。ソース・リストはエディタで，ア

センブル・リストとラベル・リストはアセンブラで出力します。

アセンブル・リストを画面に出力したときは，**リスト3-2**のような行番号は付きません。アセンブル・リストの1行は左から，

> 行番号
> アドレス（またはEQUで定義した数値）
> マシン・コード（オブジェクト）
> ラベル
> ニモニック
> オペランド

となっています。

アセンブル・リストを見る上で注意してほしいのは，オブジェクトが初めの4バイトしか出力しないことです。たとえば36行目は実際のオブジェクトは14バイトになりますが，4バイト（キャラクタ・コードのSamp）しか出力していません。

リストのラベルや擬似命令にも注目しておいてください。なお，ラベル・リストを画面に出力したときは，1行に1つのラベルを出力します。

3-7-5 エラーメッセージ

アセンブラが出力するエラーメッセージは，全部で13種類あります（**表3-2**）。ただし，アセンブルを中止しないエラーでもパス1でエラーが起きるとパス2は行われません。

表3-2 エラーメッセージ

アセンブルを中止するもの	
メッセージ	意　　味
% Object area full	オブジェクト・エリアがいっぱいになった。
% Label table full	ラベル・テーブルがいっぱいになった。

アセンブルを中止しないもの		
エラーコード	メッセージ	意　　味
A	Address Overflow	アドレスがFFFF番地を超えた。
B	Balance Error	左右のカッコの数が合わない。引用符の使いかたがおかしい。
E	Expression Error	演算子または式の記述がおかしい。
F	Format Error	オペランドの数が合わない，該当するニモニックがない，その他。
L	Label Error	ラベルに予約語を使っている。または記述がおかしい。
M	Multiply Defined Label	同じラベル名を複数個定義した。
O	Operand Error	オペランドの記述がおかしい。
P	Phase Error	パス1とパス2でラベルの値が異なる。
R	Reference Error	リラティブ・ジャンプが範囲外。インデックス・アドレッシングでディスプレイスメントが範囲外。
U	Undefined Label	未定義ラベルを参照した。
V	Value Error	オペランドに記述された値が違っている。

リスト3-1 ソース・リスト

```
;
;        Sample Program
;
START    EQU     0A000H
;  **   Constants  **
CR       EQU     13
LF       EQU     10
TIMER    EQU     9000H
;  **   IOCS  **
ACCPRT   EQU     0013H;   ; Put 1 Charactor
                          ; A = ASCII Code
;
         ORG     START
         LD      HL,MESSGE
LOOP:    LD      A,(HL)
         OR      A        ; End ?
         JR      Z,CRLF
         CALL    ACCPRT
         CALL    DELAY
         INC     HL
         JR      LOOP
;  **   Delay Loop  **
DELAY:   LD      BC,TIMER
DLOOP:   DEC     BC
         LD      A,B
         OR      C        ; IF BC <> 0 THEN GOTO 'DLOOP'
         JR      NZ,DLOOP
         RET
;  **   Put CRLF & Return to System
CRLF:    LD      A,CR
         CALL    ACCPRT
         LD      A,LF
         CALL    ACCPRT
         RET
;
MESSGE:  DEFM    'Sample Program'
         DEFB    0        ; End Maker
;
         END
```

リスト3-2 アセンブル・リスト

```
 1:                   ;
 2:                   ;        Sample Program
 3:                   ;
 4:   A000 =          START    EQU     0A000H
 5:                   ;  **   Constants  **
 6:   000D =          CR       EQU     13
 7:   000A =          LF       EQU     10
 8:   9000 =          TIMER    EQU     9000H
 9:                   ;  **   IOCS  **
10:   0013 =          ACCPRT   EQU     0013H;   ; Put 1 Charactor
11:                                             ; A = ASCII Code
12:                   ;
13:   A000                     ORG     START
14:   A000 2124A0              LD      HL,MESSGE
15:   A003 7E         LOOP:    LD      A,(HL)
16:   A004 B7                  OR      A        ; End ?
17:   A005 2812                JR      Z,CRLF
18:   A007 CD1300              CALL    ACCPRT
19:   A00A CD10A0              CALL    DELAY
20:   A00D 23                  INC     HL
21:   A00E 18F3                JR      LOOP
22:                   ;  **   Delay Loop  **
23:   A010 010090     DELAY:   LD      BC,TIMER
24:   A013 0B         DLOOP:   DEC     BC
25:   A014 78                  LD      A,B
```

```
26:    A015 B1                   OR      C        ; IF BC <> 0 THEN GOTO 'DLOOP'
27:    A016 20FB                 JR      NZ,DLOOP
28:    A018 C9                   RET
29:                    ; **    Put CRLF & Return to System
30:    A019 3E0D       CRLF:     LD      A,CR
31:    A01B CD1300               CALL    ACCPRT
32:    A01E 3E0A                 LD      A,LF
33:    A020 CD1300               CALL    ACCPRT
34:    A023 C9                   RET
35:                    ;
36:    A024 53616D70 MESSGE:    DEFM    'Sample Program'
37:    A032 00                   DEFB    0        ; End Maker
38:                    ;
39:    A033                      END
```

リスト3-3 ラベル・リスト

```
0013  ACCPRT    000D  CR       A019  CRLF     A010  DELAY
A013  DLOOP     000A  LF       A003  LOOP     A024  MESSGE
A000  START     9000  TIMER
```

エディタ・アセンブラの実行開始番地は14A0番地です。リストを打ち込む方に注意してほしいことは，わずか1ヵ所でも入力ミスがあると，アセンブルされたマシン・コードが違っていたり，エディタの誤動作をひきおこしたりすることです。

ところが，面倒なことに少々，入力ミスをしても，正常に動作しているかのごとく見える場合があるのです。正常に動いているのか，異常な動きなのか，チェックはきわめて大変で，あらゆる場合をこと細かく点検しなければなりません。

そこで，あらかじめ完成された「エディタ・アセンブラ」を希望される人のために，カセット版の『X1エディタ・アセンブラ』を用意しました(定価10,000円・送料含む)。

これはIPL起動になっています。また，本誌第4章，第5章のソースも入れてありますので，マシン語の速習にも役立ちます。御希望の方は現金書留で下記まで。

〒102 東京都千代田区紀尾井町3—20
㈱MIA「X1エディタ・アセンブラ」係

エディタ・アセンブラ

```
:14A0=C3 B5 14 C3 C5 14 C3 E6
:14A8=46 C3 19 2E 00 8E 4A C3
:14B0=D3 17 C3 A7 17 AF 32 8B
:14B8=0A CD 8C 09 CD 7A 33 21
:14C0=A3 14 22 2B 01 21 D0 14
:14C8=22 53 10 3E 50 32 06 00
:14D0=31 00 00 3E 02 32 8B 0A
:14D8=CD 8C 09 21 83 17 22 7E
:14E0=14 01 00 10 AF ED 79 04
:14E8=ED 79 04 ED 79 04 ED 79
:14F0=04 32 72 14 32 1E 00 32
:14F8=16 00 3E 4F 32 1F 00 3E
:1500=18 32 17 00 2A 90 4A 22
:1508=9F 18 21 0A 15 E5 CD A7
:1510=17 3E 0C CD 13 00 21 05
:1518=05 11 19 18 CD 13 18 2C
:1520=11 4C 18 CD 12 18 11 5B
:1528=18 CD 12 18 11 79 18 CD
:1530=12 18 E5 2A 9F 18 ED 5B
:1538=94 4A CD 33 42 28 1F 11
:1540=5A 16 CD 0B 00 23 CD 02
:1548=12 3E 2D CD 13 00 2A 94
:1550=4A CD 02 12 CD BA 04 3E
:1558=29 CD 13 00 18 06 11 67
:1560=16 CD 0B 00 E1 11 86 18
:1568=CD 0B 00 11 97 18 CD 12
:1570=18 21 06 06 22 0E 00 11
:1578=00 FF CD 03 00 D8 CD A5
:1580=15 D8 FE 53 28 0F FE 41
:1588=CA E2 15 FE 4F CA 31 16
:1590=FE 4D 28 0A C9 CD D3 17
:1598=CD 1C 2E C3 D0 14 CD D3
:15A0=17 C3 E6 46 13 1A B7 37
:15A8=C8 FE 20 28 F7 C9 E5 2E
:15B0=00 1A 13 BC 20 19 1A 13
:15B8=FE 3A 20 13 1A CD 51 14
:15C0=13 FE 58 28 0B 2C FE 43
:15C8=28 06 2C FE 50 28 01 37
:15D0=7D E1 C9 E5 21 DF 15 85
:15D8=6F 30 01 24 7E E1 C9 58
:15E0=43 50 1A B7 C8 13 FE 28
:15E8=26 F8 26 41 CD AE 15 D8
:15F0=32 75 1E CD D3 15 32 6D
:15F8=18 CD A5 15 D8 26 4C CD
```

```
:1600=AE 15 D8 32 76 1E CD D3
:1608=15 32 71 18 CD A5 15 D8
:1610=26 45 CD AE 15 D8 32 77
:1618=1E CD D3 15 32 75 18 CD
:1620=A5 15 D8 CD D3 17 3E 0C
:1628=CD 13 00 CD A6 18 C3 95
:1630=17 1A B7 C8 13 FE 3A 20
:1638=F8 CD 1F 11 D8 1A E6 DF
:1640=FE 48 C2 0A 15 3E 0F 32
:1648=0F 00 E5 FD E1 3A 8B 0A
:1650=FE 02 20 21 CD 9B 16 C3
:1658=95 17 46 72 65 65 20 41
:1660=72 65 61 20 28 20 00 4E
:1668=6F 20 46 72 65 65 20 41
:1670=72 65 61 20 00 11 7E 16
:1678=CD 0B 00 C3 95 17 0D 53
:1680=63 72 65 65 6E 20 43 6C
:1688=65 61 72 20 62 79 20 57
:1690=69 64 74 68 20 34 30 2C
:1698=38 30 00 ED 78 AF 32 9C
:16A0=18 21 00 D0 18 0A EB 3E
:16A8=2D CD 13 00 CD 02 12 EB
:16B0=CD FD 1D 23 5F CD FD 1D
:16B8=23 57 CD FD 1D 23 4F CD
:16C0=FD 1D 23 47 B1 B2 B3 C8
:16C8=E5 FD E5 E1 19 11 FC 16
:16D0=CD 0B 00 CD 02 12 EB E1
:16D8=78 B1 28 CA E5 2A 9F 18
:16E0=B7 ED 52 D4 13 17 E1 30
:16E8=1A E5 2A 94 4A B7 ED 52
:16F0=E1 38 10 CD FD 1D 12 13
:16F8=23 0B 18 DC 0D 4C 6F 61
:1700=64 20 00 CD A7 04 EB CD
:1708=02 12 11 57 17 CD 0B 00
:1710=C3 95 17 F5 3A 9C 18 B7
:1718=20 3B E5 2A 92 4A B7 ED
:1720=52 E1 30 DF CD DA 32 D5
:1728=11 64 17 CD 0B 00 D1 3E
:1730=01 CD 1B 00 CD 13 00 CD
:1738=51 14 FE 59 C2 03 17 CD
:1740=E3 32 CD 7A 33 E5 2A 90
:1748=4A 22 9F 18 3E 01 32 9C
:1750=18 E1 F1 37 C9 F1 C9 20
:1758=20 4C 6F 61 64 20 45 72
:1760=72 6F 72 00 20 20 20 20
:1768=20 20 20 44 65 73 74 72
:1770=6F 79 20 61 20 46 69 6C
:1778=65 20 28 59 2F 4E 29 20
:1780=3F 20 00 FE 49 20 08 11
:1788=F9 17 CD 0B 00 18 06 11
:1790=09 18 CD 0B 00 11 E7 17
:1798=CD 0B 00 3E 01 CD 1B 00
:17A0=FE 0D 20 F7 C3 D0 14 21
:17A8=F2 0E 36 01 23 36 00 01
:17B0=4E 00 11 F4 0E ED B0 3E
:17B8=05 06 00 21 CE 17 4E DD
:17C0=21 F2 0E DD 09 DD 36 00
:17C8=01 23 3D 20 F1 C9 06 18
:17D0=1C 20 32 21 F2 0E 11 00
:17D8=01 0E 0A 72 23 06 07 73
:17E0=23 10 FC 0D 20 F5 C9 0D
:17E8=50 72 65 73 73 20 52 65
:17F0=74 75 72 6E 20 4B 65 79
:17F8=00 0D 44 65 76 69 63 65
:1800=20 4F 66 66 6C 69 6E 65
:1808=00 0D 45 72 72 6F 72 20
:1810=3F 00 24 22 0E 00 C3 0B
:1818=00 57 65 6C 63 6F 6D 65
:1820=20 74 6F 20 58 31 20 45
:1828=64 69 74 6F 72 20 41 73
:1830=73 65 6D 62 6C 65 72 20
:1838=53 79 73 74 65 6D 20 56
:1840=31 2E 30 20 62 79 20 48
:1848=2E 57 0D 00 53 63 72 65
:1850=65 6E 20 45 64 69 74 6F
:1858=72 0D 00 41 73 73 65 6D
:1860=62 6C 65 72 20 20 20 20
:1868=20 20 28 41 3A 58 20 4C
:1870=3A 58 20 45 3A 43 29 0D
:1878=00 4F 62 6A 65 63 74 20
:1880=4C 6F 61 64 20 00 20 28
:1888=4F 66 66 73 65 74 3A 30
:1890=30 30 30 48 29 0D 00 4D
:1898=6F 6E 0D 00 2F CD D0 96
:18A0=4A 3E 01 C3 1B 00 ED 78
:18A8=AF 01 3E 01 32 74 1E ED
:18B0=73 4C 1E 31 00 FF 21 4E
:18B8=1E 01 25 00 36 00 23 0B
:18C0=78 B1 20 F8 3D 3D 32 56
:18C8=1E 21 00 C0 01 00 40 AF
:18D0=CD 32 1E 21 BC 19 CD 7C
:18D8=1D 3E 01 21 D8 19 CD 04
:18E0=1A 3A 74 1E B7 3E 01 C2
:18E8=B4 19 18 07 ED 73 4C 1E
:18F0=31 00 FF CD 22 1C 3E 02
:18F8=21 E1 19 CD 04 1A CD 73
:1900=1D 21 E9 19 CD 7C 1D 11
:1908=59 1E 2A ED 2D CD 64 24
:1910=AF 12 21 59 1E CD 7C 1D
:1918=21 F7 19 CD 7C 1D 11 59
:1920=1E 2A 6F 1E CD 2B 24 AF
:1928=12 21 59 1E CD 7C 1D 21
:1930=FA 19 CD 7C 1D CD 73 1D
:1938=3A 56 1E B7 28 0C 3C 3C
:1940=28 08 3E 0C CD D5 1D DA
:1948=4E 1A CD 73 1D 3A 76 1E
:1950=B7 28 5F 21 00 00 ED 5B
:1958=6F 1E E5 B7 ED 52 E1 30
:1960=3E E5 CD F1 1C 2A 65 1E
:1968=11 59 1E CD 64 24 3E 20
:1970=EB 77 23 77 21 65 1E 06
:1978=04 77 23 10 FC 36 00 21
:1980=59 1E 3A 76 1E 3D 20 08
:1988=CD 73 1D CD 7C 1D 18 0B
:1990=D1 D5 7B E6 03 CC 4F 1D
:1998=CD 58 1D E1 23 18 B7 3A
:19A0=76 1E 3D 20 05 CD 73 1D
:19A8=18 08 CD 4F 1D 3E 0C CD
:19B0=D5 1D 3E 02 32 73 1E ED
:19B8=7B 4C 1E C9 0D 58 31 20
:19C0=53 65 6C 66 20 41 73 73
:19C8=65 6D 62 6C 65 72 20 20
```

```
:19D0=52 65 76 20 31 2E 30 00
:19D8=0D 0D 50 61 73 73 20 31
:19E0=00 0D 50 61 73 73 20 32
:19E8=00 0D 45 6E 64 20 41 64
:19F0=64 72 65 73 73 20 00 20
:19F8=2C 00 20 4C 61 62 65 6C
:1A00=28 73 29 00 32 E0 2D CD
:1A08=7C 1D 21 00 00 22 54 1E
:1A10=CD 78 1E CD 73 1D 2A 17
:1A18=2E 7C B5 F5 28 12 E5 CD
:1A20=73 1D E1 11 59 1E CD 2B
:1A28=24 AF 12 21 59 1E 18 03
:1A30=21 41 1A CD 7C 1D 21 44
:1A38=1A CD 7C 1D F1 C8 C3 AF
:1A40=1A 4E 6F 00 20 45 72 72
:1A48=6F 72 28 73 29 00 21 54
:1A50=1A C3 AC 1A 0D 25 20 41
:1A58=62 6F 72 74 65 64 00 21
:1A60=65 1A C3 AC 1A 0D 25 20
:1A68=4F 62 6A 65 63 74 20 61
:1A70=72 65 61 20 66 75 6C 6C
:1A78=00 21 7F 1A C3 AC 1A 0D
:1A80=25 20 4C 61 62 65 6C 20
:1A88=74 61 62 6C 65 20 66 75
:1A90=6C 6C 00 21 99 1A C3 AC
:1A98=1A 0D 25 20 53 63 72 65
:1AA0=65 6E 20 6E 6F 74 20 6D
:1AA8=6F 64 65 00 CD 7C 1D CD
:1AB0=73 1D 3E FF C3 B4 19 3A
:1AB8=74 1E B7 28 08 ED 73 71
:1AC0=1E AF C3 B4 19 CD E0 1D
:1AC8=DA 4E 1A 2A 54 1E 7C B5
:1AD0=28 18 7E B7 37 C8 22 8E
:1AD8=4A CD 6B 36 D8 ED 5B F0
:1AE0=2D 22 54 1E 22 52 1E 13
:1AE8=B7 C9 2A 92 4A 11 00 00
:1AF0=7E B7 37 C8 18 EB 21 FC
:1AF8=1A 18 B1 3A 20 20 20 20
:1B00=25 20 41 73 73 65 6D 62
:1B08=6C 65 72 20 73 6F 75 72
:1B10=65 63 20 65 72 72 6F 72
:1B18=00 2A 4E 1E 23 23 E5 F5
:1B20=CD 16 1D F1 E1 E5 23 23
:1B28=D5 19 23 22 50 1E 2B CD
:1B30=24 1D D1 E1 13 C3 31 1D
:1B38=B7 28 0C 7A B3 C8 D5 AF
:1B40=CD 19 1B D1 1B 18 F4 E5
:1B48=2A 4E 1E 23 23 CD 16 1D
:1B50=7A B3 28 0D 2A 50 1E 22
:1B58=4E 1E 23 23 23 23 22 50
:1B60=1E D1 2A 4E 1E E5 CD 31
:1B68=1D E1 23 23 11 00 00 C3
:1B70=31 1D 3A 75 1E B7 C8 3D
:1B78=20 10 11 08 00 19 CD 73
:1B80=1D CD 7C 1D 2A 52 1E C3
:1B88=8B 1D E5 CD AD 1B E1 CD
:1B90=4F 1D CD 58 1D 2A 52 1E
:1B98=C3 64 1D 47 3A 77 1E FE
:1BA0=02 DA 7C 1D 04 E5 CC AD
:1BA8=1B E1 C3 58 1D 21 56 1E
:1BB0=34 7E FE 32 D8 36 00 2A
:1BB8=57 1E 7C B5 3E 0C C4 D5
:1BC0=1D DA 4E 1A 21 F1 1B CD
:1BC8=58 1D 21 BD 19 CD 58 1D
:1BD0=21 F6 1B CD 58 1D 2A 57
:1BD8=1E 23 22 57 1E 11 59 1E
:1BE0=CD 2B 24 AF 12 21 59 1E
:1BE8=CD 58 1D CD 4F 1D C3 4F
:1BF0=1D 20 20 20 20 00 20 20
:1BF8=20 20 50 41 47 45 00 E5
:1C00=CD A9 1C E1 30 19 11 06
:1C08=00 19 ED 5B 65 1E 73 23
:1C10=72 AF C9 E5 CD A9 1C E1
:1C18=38 05 CD CB 1C AF C9 F6
:1C20=FF C9 2A 6F 1E 7C B5 C8
:1C28=2B 7C B5 C8 01 00 00 50
:1C30=59 C5 D5 C5 EB 11 67 1E
:1C38=CD F4 1C C1 D1 E1 23 C5
:1C40=D5 E5 CD F1 1C 11 67 1E
:1C48=21 5F 1E CD E2 1C E1 D1
:1C50=28 13 38 11 54 5D D5 E5
:1C58=01 08 00 11 67 1E 21 5F
:1C60=1E ED B0 E1 D1 23 C1 D5
:1C68=E5 EB 2A 6F 1E 2B B7 ED
:1C70=52 E1 D1 30 CA D5 EB B7
:1C78=ED 42 D1 28 1F C5 D5 C5
:1C80=EB 11 67 1E CD F4 1C E1
:1C88=E5 CD F1 1C E1 E3 11 5F
:1C90=1E CD 01 1D E1 11 67 1E
:1C98=CD 01 1D C1 03 2A 6F 1E
:1CA0=2B 2B B7 ED 42 D2 2F 1C
:1CA8=C9 EB 01 00 00 2A 6F 1E
:1CB0=B7 ED 42 C8 60 69 C5 D5
:1CB8=CD F1 1C D1 21 5F 1E CD
:1CC0=E2 1C 28 04 C1 03 18 E5
:1CC8=C1 37 C9 E5 2A 6F 1E E5
:1CD0=11 00 02 B7 ED 52 E1 D2
:1CD8=79 1A 23 22 6F 1E 2B D1
:1CE0=18 1F D5 E5 06 06 1A 96
:1CE8=20 04 23 13 10 F8 E1 D1
:1CF0=C9 11 5F 1E 29 29 29 01
:1CF8=00 C0 09 01 08 00 C3 04
:1D00=1E 29 29 29 01 00 C0 09
:1D08=EB 01 08 00 C3 1B 1E 11
:1D10=00 D0 19 C3 FD 1D 11 00
:1D18=D0 19 CD FD 1D 5F 23 CD
:1D20=FD 1D 57 C9 CD 46 1D D2
:1D28=5F 1A 11 00 D0 19 C3 F6
:1D30=1D D5 CD 46 1D D2 5F 1A
:1D38=11 00 D0 19 D1 7B CD F6
:1D40=1D 23 7A C3 F6 1D E5 11
:1D48=FC 2F B7 ED 52 E1 C9 E5
:1D50=21 F4 1D CD 58 1D E1 C9
:1D58=7E B7 C8 CD D5 1D DA 4E
:1D60=1A 23 18 F4 7E B7 C8 FE
:1D68=0D C8 CD D5 1D DA 4E 1A
:1D70=23 18 F1 E5 21 F4 1D CD
:1D78=7C 1D E1 C9 7E B7 C8 CD
:1D80=13 00 23 CD 9D 1D DA 4E
:1D88=1A 18 F1 7E B7 C8 FE 0D
:1D90=C8 CD 13 00 23 CD 9D 1D
:1D98=DA 4E 1A 18 EE 3A 2F 00
```

```
:1DA0=CB 77 37 3F C0 3A 2E 00
:1DA8=FE 03 37 28 18 FE 20 28
:1DB0=02 B7 C9 CD C5 1D 3E 01
:1DB8=CD 1B 00 FE 03 37 28 05
:1DC0=FE 20 20 F2 B7 F5 AF 32
:1DC8=2E 00 32 36 00 32 A6 0E
:1DD0=32 A7 0E F1 C9 FE 0D 28
:1DD8=16 FE 09 28 0D CD DC 12
:1DE0=3A 36 00 FE 03 37 28 DD
:1DE8=B7 C9 CD 15 13 18 F1 CD
:1DF0=D5 12 18 EC 0D 00 C5 44
:1DF8=4D ED 79 C1 C9 C5 44 4D
:1E00=ED 78 C1 C9 7C 60 47 7D
:1E08=69 4F ED 78 12 03 13 2B
:1E10=7C B5 20 F6 7C 60 47 7D
:1E18=69 4F C9 7A 50 47 7B 59
:1E20=4F 7E ED 79 03 23 1B 7A
:1E28=B3 20 F6 7A 50 47 7B 59
:1E30=4F C9 D5 F5 57 7C 60 47
:1E38=7D 69 4F ED 51 03 2B 7C
:1E40=B5 20 F8 7C 60 47 7D 69
:1E48=4F F1 D1 C9 13 FE 0D C2
:1E50=79 34 C9 AF CD 3B 3A CD
:1E58=9E 3A C1 D1 E1 21 CC 02
:1E60=E5 D5 2A 23 40 E5 E5 2A
:1E68=FD 3F E5 21 CC 02 E5 21
:1E70=A6 34 E5 C5 C9 CD F9 34
:1E78=ED 73 9D 18 AF 32 E1 2D
:1E80=67 6F 22 ED 2D 22 17 2E
:1E88=ED 7B 9D 18 3A E0 2D 3D
:1E90=28 0C 21 F2 2D 06 18 3E
:1E98=20 77 23 10 FC 70 AF 32
:1EA0=EF 2D CD B7 1A DA DA 1F
:1EA8=ED 53 F0 2D CD E0 1F CA
:1EB0=AB 1F 22 EB 2D 7E CD 07
:1EB8=20 CD FB 1F DA D1 2C 3A
:1EC0=E0 2D 3D 28 09 2A ED 2D
:1EC8=11 FC 2D CD 64 24 CD 7F
:1ED0=24 CD 2A 20 B7 20 38 2A
:1ED8=EB 2D 7E 23 FE 3A 28 06
:1EE0=CD B1 24 C3 AB 1F 22 EB
:1EE8=2D 2A ED 2D CD 35 2C 2A
:1EF0=EB 2D CD E0 1F CA AB 1F
:1EF8=22 EB 2D 7E CD 07 20 CD
:1F00=FB 1F DA CE 2C CD 7F 24
:1F08=CD 2A 20 B7 CA CE 2C F5
:1F10=2A EB 2D 7E FE 3A CA D1
:1F18=2C B7 28 17 FE 0D 28 13
:1F20=FE 3B 28 0F FE 20 28 05
:1F28=FE 09 C2 CE 2C CD E0 1F
:1F30=22 EB 2D F1 21 AB 1F E5
:1F38=FE 80 38 15 D6 80 21 43
:1F40=1F 18 1C 13 25 32 25 CE
:1F48=2C 3B 25 6F 25 55 25 A9
:1F50=25 FE 50 D2 09 2C FE 40
:1F58=D2 EC 2B 21 69 1F 3D 5F
:1F60=16 00 19 19 7E 23 66 6F
:1F68=E9 C7 25 BA 27 BD 27 F7
:1F70=27 64 28 A1 28 E8 28 AE
:1F78=28 EB 28 F1 28 EE 28 F4
:1F80=28 3B 29 48 29 40 2B A7
:1F88=29 A9 29 AC 29 AF 29 B2
:1F90=29 B5 29 B8 29 C5 29 CB
:1F98=29 C8 29 28 2A 86 2A 82
:1FA0=2A BC 2A E6 2A 2A 2B 6B
:1FA8=2B A1 2B 3A E0 2D 3D 28
:1FB0=12 2A F0 2D 11 F3 2D CD
:1FB8=2B 24 3E 3A 12 21 F2 2D
:1FC0=CD 72 1B 2A EF 2D 26 00
:1FC8=ED 5B ED 2D 19 DA C5 2C
:1FD0=22 ED 2D 3A E1 2D B7 CA
:1FD8=88 1E ED 7B 9D 18 C9 23
:1FE0=7E B7 C8 FE 0D C8 FE 3B
:1FE8=C8 FE 20 28 03 FE 09 C0
:1FF0=23 18 ED FE 30 38 0C FE
:1FF8=3A 38 0A FE 3F 38 04 FE
:2000=5B 38 02 37 C9 B7 C9 FE
:2008=61 D8 FE 7B D0 E6 5F C9
:2010=2A EB 2D 7E FE 2C C2 CE
:2018=2C CD DF 1F 22 EB 2D C9
:2020=2A EB 2D CD E0 1F C2 CE
:2028=2C C9 3A E2 2D FE 02 38
:2030=3D FE 05 30 39 21 E3 2D
:2038=7E D6 41 FE 1A 30 2F 6F
:2040=26 00 11 70 20 19 5E 23
:2048=7E 93 28 22 16 00 21 8B
:2050=20 19 19 19 19 4F 06 03
:2058=11 E4 2D 1A BE 20 06 13
:2060=23 10 F8 7E C9 0D 28 06
:2068=04 23 10 FD 18 E8 AF C9
:2070=00 03 04 0C 14 19 19 19
:2078=1A 21 23 23 28 28 2A 31
:2080=33 33 42 49 49 49 49 49
:2088=4A 4A 4A 44 43 20 06 44
:2090=44 20 05 4E 44 20 09 49
:2098=54 20 17 41 4C 4C 1D 43
:20A0=46 20 43 50 20 20 0C 50
:20A8=44 20 56 50 44 52 57 50
:20B0=49 20 54 50 49 52 55 50
:20B8=4C 20 42 41 41 20 41 45
:20C0=43 20 0E 45 46 42 83 45
:20C8=46 4D 84 45 46 53 86 45
:20D0=46 57 85 49 20 20 47 4A
:20D8=4E 5A 1C 49 20 20 48 4E
:20E0=44 20 81 51 55 20 82 58
:20E8=20 20 04 58 58 20 40 41
:20F0=4C 54 46 4D 20 20 0F 4E
:20F8=20 20 20 4E 43 20 0D 4E
:2100=44 20 5F 4E 44 52 60 4E
:2108=49 20 5D 4E 49 52 5E 50
:2110=20 20 1A 52 20 20 1B 44
:2118=20 20 01 44 44 20 52 44
:2120=44 52 53 44 49 20 50 44
:2128=49 52 51 45 47 20 58 4F
:2130=50 20 45 52 20 20 0A 52
:2138=47 20 80 54 44 52 64 54
:2140=49 52 62 55 54 20 21 55
:2148=54 44 63 55 54 49 61 4F
:2150=50 20 03 55 53 48 02 45
:2158=53 20 19 45 54 20 1E 45
:2160=54 49 5B 45 54 4E 5C 4C
:2168=20 20 12 4C 41 20 4A 4C
```

```
:2170=43 20 10 4C 43 41 49 4C
:2178=44 20 59 52 20 20 13 52
:2180=41 20 4C 52 43 20 11 52
:2188=43 41 4B 52 44 20 5A 53
:2190=54 20 1F 42 43 20 08 43
:2198=46 20 44 45 54 20 18 4C
:21A0=41 20 14 52 41 20 15 52
:21A8=4C 20 16 55 42 20 07 4F
:21B0=52 20 0B 21 B8 21 18 14
:21B8=4E 5A 5A 20 4E 43 43 20
:21C0=50 4F 50 45 50 20 4D 20
:21C8=00 21 ED 21 3A E2 2D FE
:21D0=03 30 15 06 00 11 E3 2D
:21D8=1A BE 13 23 20 04 1A BE
:21E0=28 09 23 04 7E B7 20 ED
:21E8=3E FF C9 78 C9 42 20 43
:21F0=20 44 20 45 20 48 20 4C
:21F8=20 41 20 49 20 52 20 41
:2200=46 42 43 44 45 48 4C 53
:2208=50 49 58 49 59 00 CD 10
:2210=20 2A EB 2D 7E FE 28 28
:2218=21 CD AF 22 2A EB 2D 7E
:2220=FE 29 CA C8 2C 3A 0F 2E
:2228=3C 20 04 3E 40 18 6E 3A
:2230=10 2E B7 C2 D4 2C 3A 0F
:2238=2E C9 CD DF 1F 22 EB 2D
:2240=CD AF 22 2A EB 2D 7E FE
:2248=29 C2 C8 2C CD DF 1F CD
:2250=4D 23 C2 D4 2C 22 EB 2D
:2258=3A 0F 2E FE FF 20 05 3E
:2260=30 C3 9D 22 FE 0E 38 04
:2268=C6 12 18 31 3A 10 2E B7
:2270=C2 D4 2C 3A 0F 2E FE 0C
:2278=20 04 3E 10 18 1F FE 0A
:2280=20 04 3E 11 18 17 FE 0B
:2288=20 04 3E 12 18 0F FE 0D
:2290=20 04 3E 13 18 07 FE 01
:2298=C2 D4 2C 3E 14 32 0F 2E
:22A0=C9 CD AF 22 3A 0F 2E 3C
:22A8=C2 D4 2C 2A 0B 2E C9 AF
:22B0=32 10 2E 67 6F 22 0B 2E
:22B8=3D 32 0F 2E 2A EB 2D 7E
:22C0=FE 2B 28 74 FE 2D 28 73
:22C8=FE 27 20 05 CD E7 23 18
:22D0=3C FE 30 38 09 FE 3A 30
:22D8=05 CD 58 23 18 2F CD 07
:22E0=20 CD FB 1F 38 1B CD 7F
:22E8=24 CD C9 21 3C 28 0D 3D
:22F0=32 0F 2E 3A 10 2E B7 C2
:22F8=CB 2C 18 24 CD 5E 2C 18
:2300=0C FE 24 C2 D4 2C 23 22
:2308=EB 2D 2A ED 2D EB 2A 0B
:2310=2E 3A 10 2E B7 FA 1B 23
:2318=19 18 02 ED 52 22 0B 2E
:2320=2A EB 2D CD E0 1F 22 EB
:2328=2D CD 4A 23 C8 FE 2B 28
:2330=07 FE 2D 28 06 C3 D4 2C
:2338=3E 01 01 3E FF 32 10 2E
:2340=CD DF 1F 22 EB 2D 7E C3
:2348=C8 22 FE 29 C8 B7 C8 FE
:2350=0D C8 FE 3B C8 FE 2C C9
:2358=11 E3 2D 06 07 2A EB 2D
:2360=7E CD 07 20 CD 4A 23 28
:2368=18 FE 09 28 14 FE 20 28
:2370=10 FE 2B 28 0C FE 2D 28
:2378=08 12 13 23 10 E2 C3 DA
:2380=2C 22 EB 2D EB 2B 7E FE
:2388=44 28 05 FE 48 28 2C 23
:2390=36 00 21 00 00 11 E3 2D
:2398=1A B7 C8 FE 0D C8 D6 30
:23A0=FE 0A 30 76 44 4D 29 38
:23A8=71 29 38 6E 09 38 6B 29
:23B0=38 68 4F 06 00 09 38 62
:23B8=13 18 DD 36 00 21 00 00
:23C0=11 E3 2D 1A B7 C8 FE 0D
:23C8=C8 D6 30 FE 0A 38 08 D6
:23D0=11 FE 06 30 45 C6 0A 4F
:23D8=06 00 7C E6 F0 20 3B 29
:23E0=29 29 29 09 13 18 DC ED
:23E8=5B EB 2D 1A FE 27 20 2A
:23F0=06 03 21 00 00 13 1A B7
:23F8=CA C8 2C FE 0D CA C8 2C
:2400=FE 09 28 12 FE 20 38 12
:2408=FE 7F 28 0E FE 27 20 06
:2410=13 1A FE 27 20 07 65 6F
:2418=10 DB C3 DA 2C ED 53 EB
:2420=2D C9 7D 87 9F BC C2 D7
:2428=2C 7D C9 D5 01 10 27 CD
:2430=56 24 01 E8 03 CD 56 24
:2438=01 64 00 CD 56 24 01 0A
:2440=00 CD 56 24 7D F6 30 12
:2448=13 E1 06 04 7E FE 30 C0
:2450=36 20 23 10 F7 C9 3E 30
:2458=B7 ED 42 38 03 3C 18 F8
:2460=09 12 13 C9 7C CD 69 24
:2468=7D F5 0F 0F 0F 0F CD 72
:2470=24 F1 E6 0F C6 30 FE 3A
:2478=38 02 C6 07 12 13 C9 21
:2480=E3 2D 06 06 3E 20 77 23
:2488=10 FC 70 23 70 2A EB 2D
:2490=11 E3 2D 7E CD 07 20 CD
:2498=F3 1F 38 0D 23 4F 04 78
:24A0=FE 07 30 EF 79 12 13 18
:24A8=EA 22 EB 2D 78 32 E2 2D
:24B0=C9 2A EB 2D 7E FE 20 28
:24B8=05 FE 09 C2 D1 2C CD E0
:24C0=1F CA D1 2C 11 10 25 06
:24C8=03 7E CD 07 20 EB BE EB
:24D0=C2 CE 2C 13 23 10 F2 CD
:24D8=E0 1F CA CE 2C 22 EB 2D
:24E0=2A E3 2D E5 2A E5 2D E5
:24E8=2A E7 2D E5 CD A1 22 EB
:24F0=E1 22 E7 2D E1 22 E5 2D
:24F8=E1 22 E3 2D EB E5 11 FC
:2500=2D CD 64 24 3E 3D 32 01
:2508=2E E1 CD 35 2C C3 20 20
:2510=45 51 55 CD A1 22 CD 20
:2518=20 2A 0B 2E 22 ED 2D 3A
:2520=E0 2D 3D C8 E5 11 FC 2D
:2528=CD 64 24 E1 AF 57 5F C3
:2530=38 1B CD 20 20 3E FF 32
:2538=E1 2D C9 CD A1 22 CD 82
```

```
:2540=2C 2A EB 2D CD E0 1F C8
:2548=FE 2C C2 D4 2C CD DF 1F
:2550=22 EB 2D 18 E6 CD A1 22
:2558=CD 87 2C 2A EB 2D CD E0
:2560=1F C8 FE 2C C2 D4 2C CD
:2568=DF 1F 22 EB 2D 18 E6 2A
:2570=EB 2D 7E FE 27 C2 D4 2C
:2578=23 7E B7 CA C8 2C FE 0D
:2580=CA C8 2C FE 09 28 14 FE
:2588=20 DA D4 2C FE 7F CA D4
:2590=2C FE 27 20 06 23 7E FE
:2598=27 20 07 E5 CD 9B 2C E1
:25A0=18 D6 CD E0 1F C8 C3 CE
:25A8=2C CD A1 22 CD 20 20 ED
:25B0=5B 0B 2E CB 7A C2 DA 2C
:25B8=2A ED 2D 19 22 ED 2D 3A
:25C0=E0 2D 3D C8 C3 38 1B CD
:25C8=11 22 FE 40 CA D4 2C 2A
:25D0=0B 2E 22 0D 2E F5 CD 0E
:25D8=22 C1 4F CD 20 20 78 FE
:25E0=09 38 05 FE 10 DA 09 27
:25E8=FE 07 D2 6D 26 FE 06 20
:25F0=01 3C 87 87 87 47 79 FE
:25F8=07 30 0B FE 06 20 01 3C
:2600=C6 40 80 C3 9B 2C FE 40
:2608=20 09 3E 06 80 CD 9B 2C
:2610=C3 82 2C FE 10 20 06 3E
:2618=46 80 C3 9B 2C FE 20 28
:2620=07 FE 21 20 10 3E FD 21
:2628=3E DD C5 CD 9B 2C C1 CD
:2630=17 26 C3 8F 2C 78 FE 38
:2638=C2 D4 2C 79 FE 07 20 05
:2640=3E 57 C3 E1 26 FE 08 20
:2648=05 3E 5F C3 E1 26 FE 11
:2650=20 05 3E 0A C3 9B 2C FE
:2658=12 20 05 3E 1A C3 9B 2C
:2660=FE 30 C2 D4 2C 3E 3A CD
:2668=9B 2C C3 87 2C 79 FE 07
:2670=30 2C FE 06 20 01 3C 4F
:2678=78 FE 10 20 06 3E 70 81
:2680=C3 9B 2C FE 20 28 07 FE
:2688=21 20 41 3E FD 21 3E DD
:2690=C5 CD 9B 2C C1 CD 7D 26
:2698=2A 0D 2E C3 92 2C FE 40
:26A0=C2 6A 27 78 FE 10 20 08
:26A8=3E 36 CD 9B 2C C3 82 2C
:26B0=FE 20 28 08 FE 21 C2 D4
:26B8=2C 3E FD 21 3E DD CD 9B
:26C0=2C 3E 36 CD 9B 2C CD 98
:26C8=26 C3 82 2C 79 FE 07 C2
:26D0=D4 2C 78 FE 07 20 04 3E
:26D8=47 18 06 FE 08 20 0C 3E
:26E0=4F F5 3E ED CD 9B 2C F1
:26E8=C3 9B 2C FE 11 20 05 3E
:26F0=02 C3 9B 2C FE 12 20 05
:26F8=3E 12 C3 9B 2C FE 30 C2
:2700=D4 2C 3E 32 CD 9B 2C 18
:2708=71 79 FE 40 20 1F 78 D6
:2710=0A FE 04 30 07 87 87 87
:2718=87 3C 18 0B 3E DD 28 02
:2720=3E FD CD 9B 2C 3E 21 CD
:2728=9B 2C C3 87 2C FE 30 20
:2730=18 78 FE 0C 20 08 3E 2A
:2738=CD 9B 2C C3 87 2C D6 0A
:2740=FE 06 D2 D4 2C 16 08 18
:2748=49 78 FE 0D C2 D4 2C 79
:2750=FE 0C 28 11 FE 0E 28 08
:2758=FE 0F C2 D4 2C 3E FD 01
:2760=3E DD CD 9B 2C 3E F9 C3
:2768=9B 2C 78 FE 30 C2 D4 2C
:2770=79 FE 0C 20 0E 3E 22 CD
:2778=9B 2C 2A 0D 2E 22 0B 2E
:2780=C3 87 2C D6 0A FE 06 D2
:2788=D4 2C 16 00 2A 0D 2E 22
:2790=0B 2E FE 04 30 10 87 87
:2798=87 87 82 C6 43 F5 3E ED
:27A0=CD 9B 2C F1 18 0E 3E DD
:27A8=28 02 3E FD D5 CD 9B 2C
:27B0=D1 3E 22 82 CD 9B 2C C3
:27B8=87 2C 3E 04 FE AF F5 CD
:27C0=11 22 CD 20 20 D1 3A 0F
:27C8=2E FE 09 28 14 FE 10 D6
:27D0=0A FE 03 CA D4 2C FE 04
:27D8=30 0D 87 87 87 87 C6 C1
:27E0=01 3E F1 82 C3 9B 2C 3E
:27E8=DD 28 02 3E FD D5 CD 9B
:27F0=2C F1 C6 E1 C3 9B 2C CD
:27F8=11 22 CD 10 20 3A 0F 2E
:2800=FE 09 20 1A 16 41 CD 5A
:2808=28 16 46 CD 5A 28 16 27
:2810=CD 5A 28 CD E0 1F C2 D4
:2818=2C 3E 08 C3 9B 2C F5 CD
:2820=11 22 CD 20 20 C1 3A 0F
:2828=2E 4F 78 FE 0B 20 0B 79
:2830=FE 0C C2 D4 2C 3E EB C3
:2838=9B 2C FE 13 C2 D4 2C 79
:2840=FE 0C 28 11 FE 0E 28 08
:2848=FE 0F C2 D4 2C 3E FD 01
:2850=3E DD CD 9B 2C 3E E3 C3
:2858=9B 2C 7E CD 07 20 BA C2
:2860=D4 2C 23 C9 CD D8 28 20
:2868=05 06 80 C3 FF 28 FE 0C
:2870=28 1E FE 0E 28 08 FE 0F
:2878=C2 D4 2C 3E FD 21 3E DD
:2880=C5 CD 9B 2C C1 79 FE 0C
:2888=CA D4 2C B8 20 02 0E 0C
:2890=79 D6 0A FE 04 D2 D4 2C
:2898=87 87 87 87 C6 09 C3 9B
:28A0=2C CD D8 28 20 04 06 88
:28A8=18 55 06 08 18 0C CD D8
:28B0=28 20 05 06 98 C3 FF 28
:28B8=06 00 FE 0C C2 D4 2C 79
:28C0=D6 0A FE 04 D2 D4 2C 87
:28C8=87 87 87 80 C6 42 F5 3E
:28D0=ED CD 9B 2C F1 C3 9B 2C
:28D8=CD 11 22 F5 CD 0E 22 C1
:28E0=4F CD 20 20 78 FE 06 C9
:28E8=3E 90 01 3E A0 01 3E A8
:28F0=01 3E B0 01 3E B8 F5 CD
:28F8=11 22 C1 4F CD 20 20 79
:2900=FE 07 30 07 FE 06 20 01
:2908=3C 18 2C FE 10 28 26 FE
```

```
:2910=20 28 12 FE 21 28 11 FE
:2918=40 C2 D4 2C 3E 46 80 CD
:2920=9B 2C C3 82 2C 3E DD 21
:2928=3E FD C5 CD 9B 2C C1 CD
:2930=35 29 C3 8F 2C 3E 06 80
:2938=C3 9B 2C CD 11 22 F5 CD
:2940=20 20 F1 01 03 04 18 0B
:2948=CD 11 22 F5 CD 20 20 F1
:2950=01 0B 05 FE 07 30 0C FE
:2958=06 20 01 3C 87 87 87 80
:2960=C3 9B 2C FE 10 28 18 FE
:2968=20 28 07 FE 21 20 16 3E
:2970=FD 21 3E DD C5 CD 9B 2C
:2978=C1 CD 7F 29 C3 8F 2C 3E
:2980=30 80 C3 9B 2C D6 0A FE
:2988=06 D2 D4 2C FE 04 30 06
:2990=87 87 87 87 18 0D 3E DD
:2998=28 02 3E FD C5 CD 9B 2C
:29A0=C1 3E 20 81 C3 9B 2C AF
:29A8=01 3E 01 01 3E 02 01 3E
:29B0=03 01 3E 04 01 3E 05 01
:29B8=3E 07 F5 CD 11 22 C1 4F
:29C0=CD 20 20 18 24 3E 08 01
:29C8=3E 10 01 3E 18 F5 CD 11
:29D0=22 C1 FE 40 C2 D4 2C 2A
:29D8=0B 2E 7D E6 F8 B4 C2 D4
:29E0=2C 78 85 F5 CD 0E 22 C1
:29E8=4F 79 FE 20 28 07 FE 21
:29F0=20 16 3E FD 21 3E DD C5
:29F8=CD 9B 2C 3E CB CD 9B 2C
:2A00=CD 8F 2C C1 0E 06 18 18
:2A08=0E 06 FE 10 28 0B FE 07
:2A10=D2 D4 2C FE 06 20 01 3C
:2A18=4F C5 3E CB CD 9B 2C C1
:2A20=78 87 87 87 81 C3 9B 2C
:2A28=2A EB 2D 7E FE 28 28 27
:2A30=E5 CD 04 2B 38 0D 3C CA
:2A38=D4 2C 3D 87 87 87 C6 C2
:2A40=E1 18 06 E1 22 EB 2D 3E
:2A48=C3 F5 CD A1 22 CD 20 20
:2A50=F1 CD 9B 2C C3 87 2C CD
:2A58=11 22 F5 CD 20 20 F1 FE
:2A60=10 28 1A FE 20 28 08 FE
:2A68=21 C2 D4 2C 06 FD 21 06
:2A70=DD 2A 0B 2E 7C B5 C2 D4
:2A78=2C 78 CD 9B 2C 3E E9 C3
:2A80=9B 2C 3E 10 18 1C 2A EB
:2A88=2D E5 CD 04 2B 38 0D FE
:2A90=04 D2 D4 2C 87 87 87 C6
:2A98=20 E1 18 06 E1 22 EB 2D
:2AA0=3E 18 F5 CD A1 22 CD 20
:2AA8=20 F1 CD 9B 2C 2A 0B 2E
:2AB0=ED 5B ED 2D B7 ED 52 2B
:2AB8=2B C3 92 2C 2A EB 2D E5
:2AC0=CD 04 2B 38 0D 3C CA D4
:2AC8=2C 3D 87 87 87 C6 C4 E1
:2AD0=18 06 E1 22 EB 2D 3E CD
:2AD8=F5 CD A1 22 CD 20 20 F1
:2AE0=CD 9B 2C C3 87 2C 2A EB
:2AE8=2D CD E0 1F 3E C9 CA 9B
:2AF0=2C CD 04 2B D2 CE 2C 3C
:2AF8=CA D4 2C 3D 87 87 87 C6
:2B00=C0 C3 9B 2C CD 7F 24 CD
:2B08=B3 21 47 2A EB 2D CD E0
:2B10=1F 22 EB 2D 28 11 FE 2C
:2B18=C2 CE 2C CD DF 1F 22 EB
:2B20=2D CA D4 2C 78 B7 C9 78
:2B28=37 C9 CD A1 22 CD 20 20
:2B30=2A 0B 2E 7D E6 C7 B4 C2
:2B38=D4 2C 3E C7 B5 C3 9B 2C
:2B40=CD A1 22 CD 20 20 2A 0B
:2B48=2E 7C B7 C2 D4 2C 7D FE
:2B50=03 D2 D4 2C F5 3E ED CD
:2B58=9B 2C F1 06 46 B7 28 07
:2B60=06 56 3D 28 02 06 5E 78
:2B68=C3 9B 2C CD D8 28 20 17
:2B70=04 79 FE 30 20 17 3E DB
:2B78=CD 9B 2C 2A 0B 2E 7C B7
:2B80=C2 D4 2C 7D C3 9B 2C FE
:2B88=06 D2 D4 2C 79 FE 14 C2
:2B90=D4 2C C5 3E ED CD 9B 2C
:2B98=F1 87 87 87 C6 40 C3 9B
:2BA0=2C CD 11 22 2A 0B 2E E5
:2BA8=F5 CD 0E 22 C1 4F CD 20
:2BB0=20 E1 22 0B 2E 79 FE 06
:2BB8=20 17 0C 78 FE 30 20 17
:2BC0=3E D3 CD 9B 2C 2A 0B 2E
:2BC8=7C B7 C2 D4 2C 7D C3 9B
:2BD0=2C FE 06 D2 D4 2C 78 FE
:2BD8=14 C2 D4 2C C5 3E ED CD
:2BE0=9B 2C C1 79 87 87 87 C6
:2BE8=41 C3 9B 2C D6 40 5F 16
:2BF0=00 21 FC 2B 19 7E CD 9B
:2BF8=2C C3 20 20 D9 27 2F 3F
:2C00=37 00 76 F3 FB 07 17 0F
:2C08=1F D6 50 5F 16 00 21 20
:2C10=2C 19 7E F5 3E ED CD 9B
:2C18=2C F1 CD 9B 2C C3 20 20
:2C20=A0 B0 A8 B8 A1 B1 A9 B9
:2C28=44 6F 67 4D 45 A2 B2 AA
:2C30=BA A3 B3 AB BB 22 E9 2D
:2C38=3A E0 2D 3D 20 0C 21 E3
:2C40=2D CD 13 1C B7 C8 3E 0A
:2C48=18 11 E5 21 E3 2D CD FF
:2C50=1B 2A E9 2D D1 B7 ED 52
:2C58=C8 3E 0E C3 9E 2D 21 E3
:2C60=2D 3A E0 2D 3D 20 0A CD
:2C68=FF 1B 2A E9 2D B7 C8 18
:2C70=0D CD FF 1B 2A E9 2D B7
:2C78=C8 3E 12 CD 9E 2D 21 00
:2C80=00 C9 3A 0B 2E 18 14 CD
:2C88=82 2C 3A 0C 2E 18 0C 2A
:2C90=0B 2E 3A E0 2D 3D 28 03
:2C98=CD 22 24 47 3A E0 2D 3D
:2CA0=28 18 3A EF 2D FE 04 30
:2CA8=0D 87 5F 16 00 21 01 2E
:2CB0=19 EB 78 CD 69 24 78 CD
:2CB8=19 1B 21 EF 2D 34 7E FE
:2CC0=80 D2 CE 2C C9 3E 00 01
:2CC8=3E 02 01 3E 04 01 3E 06
:2CD0=01 3E 08 01 3E 0C 01 3E
:2CD8=10 01 3E 14 CD 9E 2D C3
```

```
:2CE0=AB 1F F8 2C 09 2D 17 2D
:2CE8=28 2D 35 2D 41 2D 58 2D
:2CF0=66 2D 72 2D 82 2D 92 2D
:2CF8=41 64 64 72 65 73 73 20
:2D00=4F 76 65 72 66 6C 6F 77
:2D08=00 42 61 6C 61 6E 63 65
:2D10=20 45 72 72 6F 72 00 45
:2D18=78 70 72 65 73 73 69 6F
:2D20=6E 20 45 72 72 6F 72 00
:2D28=46 6F 72 6D 61 74 20 45
:2D30=72 72 6F 72 00 4C 61 62
:2D38=65 6C 20 45 72 72 6F 72
:2D40=00 4D 75 6C 74 69 70 6C
:2D48=79 20 44 65 66 69 6E 65
:2D50=64 20 4C 61 62 65 6C 00
:2D58=4F 70 65 72 61 6E 64 20
:2D60=45 72 72 6F 72 00 50 68
:2D68=61 73 65 20 45 72 72 6F
:2D70=72 00 52 65 66 65 72 65
:2D78=6E 63 65 20 45 72 72 6F
:2D80=72 00 55 6E 64 65 66 69
:2D88=6E 65 64 20 4C 61 62 65
:2D90=6C 00 56 61 6C 75 65 20
:2D98=45 72 72 6F 72 00 6F 26
:2DA0=00 11 E2 2C 19 5E 23 56
:2DA8=1A D5 32 FA 2D 21 D9 2D
:2DB0=3E FF CD 9B 1B 2A F0 2D
:2DB8=11 11 2E CD 2B 24 AF 12
:2DC0=21 11 2E CD 9B 1B 21 DC
:2DC8=2D AF CD 9B 1B 2A 17 2E
:2DD0=23 22 17 2E E1 AF C3 9B
:2DD8=1B 0D 20 00 3A 20 20 00
:2DE0=00 00 00 00 00 00 00 00
:2DE8=00 00 00 00 00 00 00 00
:2DF0=00 00 00 00 00 00 00 00
:2DF8=00 00 00 00 00 00 00 00
:2E00=00 00 00 00 00 00 00 00
:2E08=00 00 00 00 00 00 00 00
:2E10=00 00 00 00 2C 00 65 66
:2E18=73 CD 7A 33 ED 73 52 37
:2E20=DD 21 71 42 CD 8C 09 CD
:2E28=8B 33 CD CA 32 2A 92 4A
:2E30=22 8E 4A CD 49 33 CD EC
:2E38=32 DD CB 00 C6 CD 83 32
:2E40=DD CB 00 4E C4 A2 33 21
:2E48=01 00 22 4A 37 31 00 FF
:2E50=3E 01 32 4E 37 CD 3F 37
:2E58=CD B1 36 21 D9 36 E5 CD
:2E60=DA 32 21 05 00 22 0E 00
:2E68=21 00 FF 11 EF 2E CD E7
:2E70=2E EB 3A 48 37 3C 6F 26
:2E78=00 CD 81 2F EB 11 F8 2E
:2E80=CD E7 2E EB 2A 4A 37 CD
:2E88=81 2F EB 11 FF 2E CD E7
:2E90=2E EB 2A 58 37 CD 81 2F
:2E98=EB 36 00 11 00 FF CD 0B
:2EA0=00 3A 0E 00 47 3E 23 90
:2EA8=38 06 47 CD BA 04 10 FB
:2EB0=CD E3 32 3E 01 CD 1B 00
:2EB8=4F FE 20 38 6D DD CB 00
:2EC0=4E C0 CD 36 34 28 1B CD
```

```
:2EC8=08 2F DD CB 00 4E C0 CD
:2ED0=7C 2F CD F1 33 C8 CD DA
:2ED8=32 2A 8E 4A CD 06 33 C3
:2EE0=E3 32 0E 0D C3 29 32 1A
:2EE8=B7 C8 13 77 23 18 F8 43
:2EF0=4F 4C 3D 00 20 59 3D 00
:2EF8=20 4C 49 4E 45 3D 00 20
:2F00=55 4E 55 53 45 44 20 00
:2F08=CD F1 33 CA E0 34 DD CB
:2F10=00 46 C2 9A 34 FE 0D 28
:2F18=0A 79 FE 0D C8 FE 09 C8
:2F20=C3 F8 34 79 FE 0D C2 9A
:2F28=34 C9 21 37 2F 87 5F 16
:2F30=00 19 7E 23 66 6F E9 80
:2F38=2F 79 30 80 2F 95 31 C7
:2F40=2F 1F 30 AF 30 DD 31 EC
:2F48=2F A4 31 EF 42 F0 3B 25
:2F50=3A 29 32 80 2F 80 2F 75
:2F58=40 5A 37 71 31 EC 2F 1C
:2F60=32 80 2F 73 32 06 31 44
:2F68=30 FD 31 3C 31 80 2F C7
:2F70=2F EC 2F 1F 30 44 30 ED
:2F78=7B 52 37 C9 79 CD 13 00
:2F80=C9 3E 01 32 C6 2F 01 10
:2F88=27 CD A4 2F 01 E8 03 CD
:2F90=A4 2F 01 64 00 CD A4 2F
:2F98=01 0A 00 CD A4 2F 7D F6
:2FA0=30 12 13 C9 3E 30 B7 ED
:2FA8=42 38 03 3C 18 F8 09 FE
:2FB0=30 28 07 12 13 AF 32 C6
:2FB8=2F C9 F5 3A C6 2F B7 28
:2FC0=02 F1 C9 F1 18 ED 2D CD
:2FC8=F1 33 C8 CD 97 36 FE 0D
:2FD0=C2 E9 35 CD 4E 34 CD C2
:2FD8=32 C0 CD DA 32 2A 8E 4A
:2FE0=CD F8 30 CD E3 32 C9 3E
:2FE8=1C C3 13 00 2A 8E 4A CD
:2FF0=FA 33 D8 CD A4 36 CD E8
:2FF8=33 C2 E9 35 CD 59 34 28
:3000=07 3D 32 0F 00 C3 E9 35
:3008=CD 3F 36 CD 16 30 C3 EC
:3010=35 3E 1D C3 13 00 3E 0F
:3018=CD 13 00 CD F8 30 C9 CD
:3020=3F 36 D8 EB CD 42 36 CD
:3028=59 34 28 07 3D 32 0F 00
:3030=C3 AE 35 CD DA 32 CD 16
:3038=30 CD E3 32 C3 AE 35 3E
:3040=1E C3 13 00 CD F1 33 C8
:3048=CD 6B 36 38 0C CD 4E 34
:3050=28 13 3C 32 0F 00 C3 AE
:3058=35 2B 7E FE 0D C0 23 22
:3060=8E 4A C3 C2 32 CD DA 32
:3068=CD C2 32 CD F8 30 CD E3
:3070=32 C3 AE 35 3E 1F C3 13
:3078=00 2A 8E 4A CD D7 30 38
:3080=0E 2B CD FA 33 DA A9 30
:3088=CD D7 30 28 16 30 F2 2B
:3090=CD FA 33 DA A9 30 CD D7
:3098=30 28 08 38 F2 22 8E 4A
:30A0=C3 E9 35 22 8E 4A C3 FC
:30A8=2F 22 8E 4A C3 EC 32 CD
```

```
:30B0=F1 33 C8 FE 0D 20 06 CD
:30B8=97 36 C3 D3 2F 2A 8E 4A
:30C0=CD D7 30 28 0C 23 30 F8
:30C8=CD D7 30 28 04 23 38 F8
:30D0=2B 22 8E 4A C3 E9 35 7E
:30D8=B7 C8 FE 0D C8 FE 30 38
:30E0=13 FE 3A 38 0C FE 80 30
:30E8=08 E6 1F 28 07 FE 1B 30
:30F0=03 F6 FF C9 F6 FF 37 C9
:30F8=F5 E5 3A 1E 00 32 0E 00
:3100=CD 06 33 E1 F1 C9 CD 4B
:3108=35 D8 3E 01 32 51 37 CD
:3110=DA 32 CD EC 32 CD 55 35
:3118=CD 16 30 3A 49 37 47 3A
:3120=17 00 B8 CA 2E 31 04 78
:3128=32 49 37 C3 E3 32 CD 27
:3130=31 CD 3F 36 D8 EB CD 42
:3138=36 C3 AE 35 CD F1 33 C8
:3140=CD 65 35 38 25 CD DA 32
:3148=3A 17 00 32 0F 00 CD C2
:3150=32 CD F8 30 3A 49 37 47
:3158=3A 16 00 B8 28 03 05 18
:3160=C6 CD 27 31 CD 6B 36 C3
:3168=AE 35 2B 7E FE 0D C0 18
:3170=D4 CD 4B 35 D8 22 8E 4A
:3178=CD FB 32 32 51 37 CD 55
:3180=35 22 8E 4A E5 CD DA 32
:3188=CD EC 32 CD 49 33 CD E3
:3190=32 E1 C3 84 35 CD 4B 35
:3198=CD FB 32 32 51 37 CD 25
:31A0=35 D8 18 DD DD CB 00 4E
:31A8=C0 2A 8E 4A DD CB 00 46
:31B0=20 09 7E FE 09 28 15 FE
:31B8=0E 30 11 E5 CD 08 2F E1
:31C0=CD 41 34 DA E2 2E CD 06
:31C8=33 C3 E9 35 CD 41 34 DA
:31D0=E2 2E 3E 09 CD 13 00 CD
:31D8=3F 36 C3 AE 35 CD F1 33
:31E0=C8 F5 CD 64 34 F1 2A 8E
:31E8=4A CD DA 32 FE 0D 28 06
:31F0=CD 06 33 C3 E3 32 CD 49
:31F8=33 CD E3 32 C9 AF 32 4E
:3200=37 CD 20 36 22 8E 4A E5
:3208=CD 6C 34 CD DA 32 AF 32
:3210=0E 00 CD 49 33 CD E3 32
:3218=E1 C3 AE 35 CD F1 33 C8
:3220=CD 31 36 CD 6C 34 C3 D2
:3228=32 DD CB 00 4E C0 CD 08
:3230=2F DD CB 00 46 28 1D CD
:3238=F1 33 CA C2 32 CD D2 32
:3240=CD C2 32 2A 8E 4A CD DA
:3248=32 3E 0F CD 13 00 CD 06
:3250=33 C3 E3 32 CD 6B 36 22
:3258=8E 4A 2B 7E FE 0D 20 08
:3260=CD 4E 34 28 DB C3 C2 32
:3268=0E 0D CD E0 34 CD C2 32
:3270=C3 D2 32 DD CB 00 46 28
:3278=06 DD CB 00 86 18 04 DD
:3280=CB 00 C6 CD DA 32 21 34
:3288=00 22 0E 00 DD CB 00 46
:3290=20 05 11 B8 32 18 0B 11
:3298=AE 32 3A 26 00 CB DF 32
:32A0=26 00 CD 0B 00 3E 07 32
:32A8=26 00 CD E3 32 C9 49 4E
:32B0=53 45 52 54 20 4F 4E 00
:32B8=20 20 20 20 20 20 20 20
:32C0=20 00 F5 3E 0D CD 13 00
:32C8=F1 C9 F5 3E 0C CD 13 00
:32D0=F1 C9 F5 3E 05 CD 13 00
:32D8=F1 C9 E5 2A 0E 00 22 48
:32E0=37 E1 C9 E5 2A 48 37 22
:32E8=0E 00 E1 C9 F5 3A 1E 00
:32F0=32 0E 00 3A 16 00 32 0F
:32F8=00 F1 C9 3A 16 00 47 3A
:3300=17 00 90 CB 2F C9 CD D2
:3308=32 CD 36 34 28 2B 7E B7
:3310=28 25 FE 0D 28 1F FE 1C
:3318=28 15 FE 1D 28 11 FE 09
:3320=20 07 CD 41 34 38 12 3E
:3328=09 CD 13 00 23 18 DA CD
:3330=C8 04 23 18 D4 B7 C9 37
:3338=C9 F5 3A 1F 00 3D 32 0E
:3340=00 3E 1C CD C8 04 F1 B7
:3348=C9 C5 D5 E5 3A 0F 00 47
:3350=3A 17 00 3C 90 47 18 03
:3358=CD 13 00 CD 06 33 38 0B
:3360=7E 23 B7 28 06 FE 0D 20
:3368=F7 10 ED CD 72 33 E1 D1
:3370=C1 C9 F5 3E 1A CD 13 00
:3378=F1 C9 E5 F5 AF 32 71 42
:3380=2A 92 4A 36 00 22 90 4A
:3388=F1 E1 C9 F5 3E 00 32 1E
:3390=00 3E 01 32 16 00 3E 4F
:3398=32 1F 00 3E 18 32 17 00
:33A0=F1 C9 D5 E5 CD DA 32 11
:33A8=D0 33 3A 26 00 CB E7 32
:33B0=26 00 21 28 00 22 0E 00
:33B8=CD 0B 00 3E 07 32 26 00
:33C0=CD E3 32 E1 D1 C9 D5 E5
:33C8=CD DA 32 11 DC 33 18 E2
:33D0=42 75 66 66 65 72 20 46
:33D8=75 6C 6C 00 20 20 20 20
:33E0=20 20 20 20 20 20 20 00
:33E8=E5 2A 8E 4A 7E FE 0D E1
:33F0=C9 E5 2A 8E 4A 7E FE 00
:33F8=E1 C9 D5 E5 ED 5B 92 4A
:3400=B7 ED 52 E1 38 04 28 02
:3408=D1 C9 EB D1 37 C9 D5 ED
:3410=5B 94 4A E5 B7 ED 52 E1
:3418=28 02 30 EE B7 D1 C9 C5
:3420=47 3A 1E 00 B8 28 02 30
:3428=06 3A 1F 00 B8 30 03 37
:3430=C1 C9 78 B7 C1 C9 C5 3A
:3438=1F 00 47 3A 0E 00 B8 C1
:3440=C9 C5 3A 0E 00 47 3A 1F
:3448=00 D6 08 B8 C1 C9 C5 3A
:3450=17 00 47 3A 0F 00 B8 C1
:3458=C9 C5 3A 16 00 47 3A 0F
:3460=00 B8 C1 C9 E5 21 01 00
:3468=22 4F 37 E1 C5 D5 E5 F5
:3470=2A 8E 4A E5 ED 5B 4F 37
:3478=19 CD 0E 34 EB CD 8B 36
```

```
:3480=E1 EB ED B0 1B ED 53 90
:3488=4A DD CB 00 4E CA F3 34
:3490=DD CB 00 8E CD C6 33 C3
:3498=F3 34 C5 D5 E5 F5 21 01
:34A0=00 22 4F 37 CD B4 34 DD
:34A8=CB 00 4E C2 F3 34 CD F8
:34B0=34 C3 F3 34 C5 D5 E5 F5
:34B8=CD 87 36 E5 ED 5B 4F 37
:34C0=CD C7 36 38 10 19 CD 0E
:34C8=34 38 0A 22 90 4A EB E1
:34D0=ED B8 C3 F3 34 E1 DD CB
:34D8=00 CE CD A2 33 C3 F3 34
:34E0=C5 D5 E5 F5 CD F8 34 22
:34E8=90 4A 36 00 DD CB 00 4E
:34F0=C4 A2 33 F1 E1 D1 C1 C9
:34F8=2A 8E 4A 71 23 CD 0E 34
:3500=22 8E 4A D0 DD CB 00 CE
:3508=C9 7E B7 28 15 FE 09 28
:3510=07 FE 0D 28 0D 1C 18 06
:3518=7B E6 F8 C6 08 5F 7B 23
:3520=B7 C9 7B 37 C9 EB CD 8B
:3528=36 EB CD 6F 36 D8 CD 43
:3530=35 20 F6 C9 C5 3A 16 00
:3538=47 3A 0F 00 90 00 32 51
:3540=37 C1 C9 3A 51 37 3D 32
:3548=51 37 C9 CD 3F 36 38 11
:3550=CD 34 35 28 0C EB CD 42
:3558=36 38 06 CD 43 35 20 F5
:3560=B7 22 54 37 C9 CD 4B 35
:3568=3A 16 00 47 3A 17 00 90
:3570=3C 32 51 37 CD 25 35 22
:3578=56 37 C9 3A 16 00 47 3A
:3580=0F 00 90 C9 CD 7B 35 28
:3588=25 32 51 37 CD 25 35 30
:3590=1D 22 8E 4A 3A 51 37 4F
:3598=3A 0F 00 91 80 32 0F 00
:35A0=2B 7E FE 0D C2 E9 35 3A
:35A8=1E 00 32 0E 00 C9 3A 0E
:35B0=00 4F 3A 1E 00 B9 28 0A
:35B8=1E 00 CD 09 35 38 03 B9
:35C0=38 F8 CD 1F 34 32 0E 00
:35C8=22 8E 4A C9 1C 18 0F ED
:35D0=5B 54 37 2A 8E 4A CD 8E
:35D8=36 EB 3A 16 00 5F 3E 0D
:35E0=ED B1 EA CC 35 7B 32 0F
:35E8=00 CD 3F 36 ED 5B 8E 4A
:35F0=EB E5 B7 ED 52 44 4D E1
:35F8=EB 28 14 1E 00 CD 09 35
:3600=32 0E 00 CD 36 34 30 0E
:3608=0B 78 B1 20 F0 2B C9 3A
:3610=1E 00 32 0E 00 C9 3A 1F
:3618=00 32 0E 00 22 8E 4A C9
:3620=CD 3F 36 EB D5 CD 6B 36
:3628=D1 B7 ED 52 22 4F 37 EB
:3630=C9 ED 5B 8E 4A D5 CD 6B
:3638=36 D1 38 ED 2B 18 EA 2A
:3640=8E 4A CD FA 33 D8 7E FE
:3648=0D 20 06 2B 7E FE 0D 28
:3650=14 ED 5B 92 4A CD 8E 36
:3658=3E 0D ED B9 28 05 2A 92
:3660=4A 37 C9 B7 23 54 5D 23
:3668=C3 FA 33 CD 87 36 EB 3E
:3670=0D ED B1 28 05 2A 90 4A
:3678=37 C9 B7 54 5D C9 2A 8E
:3680=4A ED 5B 92 4A 18 07 ED
:3688=5B 8E 4A 2A 90 4A E5 B7
:3690=ED 52 23 44 4D E1 C9 E5
:3698=2A 8E 4A 23 CD 0E 34 22
:36A0=8E 4A E1 C9 E5 2A 8E 4A
:36A8=2B CD FA 33 22 8E 4A E1
:36B0=C9 C5 D5 E5 ED 5B 90 4A
:36B8=13 2A 94 4A CD 8E 36 ED
:36C0=43 58 37 E1 D1 C1 C9 E5
:36C8=D5 CD B1 36 2A 58 37 ED
:36D0=5B 4F 37 CD 33 42 D1 E1
:36D8=C9 3A 4E 37 B7 CA 4D 2E
:36E0=FE 02 28 06 CD F0 36 C3
:36E8=4D 2E CD 29 37 C3 4D 2E
:36F0=2A 8E 4A ED 5B 4C 37 CD
:36F8=33 42 C8 38 28 3E 01 F5
:3700=CD 8E 36 EB 11 00 00 CD
:3708=1A 37 F1 2A 4A 37 B7 20
:3710=04 ED 52 18 01 19 22 4A
:3718=37 C9 3E 0D 18 01 13 ED
:3720=B1 EA 1E 37 C9 AF EB 18
:3728=D6 2A 8E 4A ED 5B 92 4A
:3730=CD 8E 36 EB 11 01 00 CD
:3738=1A 37 ED 53 4A 37 C9 E5
:3740=2A 8E 4A 22 4C 37 E1 C9
:3748=4D 3E 20 CD 76 4C 7E E6
:3750=07 C2 7A 4D C3 8B 4D CD
:3758=05 00 CD 0A 41 3E 5E CD
:3760=13 00 3E 40 81 CD 13 00
:3768=21 FB 40 E5 CD CD 40 CD
:3770=E3 32 FE 52 CA 96 37 FE
:3778=43 CA AE 37 FE 46 CA 12
:3780=38 FE 41 CA 37 3A FE 42
:3788=CA C6 37 FE 4B CA CC 37
:3790=FE 4D CA D0 14 C9 CD 65
:3798=35 2A 92 4A 22 8E 4A CD
:37A0=EB 41 D2 EC 32 CD EC 32
:37A8=CD 49 33 C3 EC 32 CD 65
:37B0=35 2A 90 4A 22 8E 4A CD
:37B8=EB 41 D2 CF 35 3A 17 00
:37C0=32 0F 00 C3 95 41 FD 21
:37C8=E7 42 18 04 FD 21 EA 42
:37D0=CD 65 35 2A 8E 4A 7E FD
:37D8=BE FF C8 CD C4 3C 38 06
:37E0=22 8E 4A C3 47 38 11 EC
:37E8=37 C3 4B 41 20 2A 2A 20
:37F0=4D 61 72 6B 65 72 20 4E
:37F8=6F 74 20 46 6F 75 6E 64
:3800=20 2A 2A 20 50 72 65 73
:3808=73 20 52 65 74 75 72 6E
:3810=1A 00 CD 3F 41 DD CB 00
:3818=AE CD 7D 38 DA 70 41 B7
:3820=CA 70 41 CD 77 39 DA 70
:3828=41 DD CB 00 FE 3A 72 42
:3830=32 73 42 2A 8E 4A CD 50
:3838=38 DA 53 41 DD 35 02 20
:3840=F5 22 8E 4A CD 8B 33 CD
:3848=EB 41 DA 8F 41 C3 98 41
```

```
:3850=EB CD 8B 36 EB 11 7D 42
:3858=1A 13 ED B1 28 02 37 C9
:3860=1A 13 B7 C8 BE 23 0B F5
:3868=78 B1 28 07 F1 28 F1 2B
:3870=03 18 E2 F1 28 04 F6 FF
:3878=37 C9 F6 FF C9 21 00 01
:3880=11 C9 38 CD 13 18 21 7D
:3888=42 06 31 0E 00 3E 01 CD
:3890=1B 00 FE 03 28 20 FE 08
:3898=28 20 FE 09 28 08 FE 0D
:38A0=28 0E FE 20 38 E7 CD C8
:38A8=04 77 23 0C 78 B9 20 DD
:38B0=36 00 F6 FF 79 C9 F6 FF
:38B8=37 C9 CD BF 38 18 CE 79
:38C0=B7 C8 0D 2B 3E 08 C3 13
:38C8=00 46 69 6E 64 3F 20 00
:38D0=0D 52 65 70 6C 61 63 65
:38D8=20 77 69 74 68 3F 20 00
:38E0=0D 4F 70 74 69 6F 6E 73
:38E8=3F 20 28 3F 20 46 6F 72
:38F0=20 49 6E 66 6F 29 20 00
:38F8=20 20 20 4E 6F 72 6D 61
:3900=6C 6C 79 20 50 72 65 73
:3908=73 20 52 65 74 75 72 6E
:3910=20 6F 6E 6C 79 2C 6F 72
:3918=20 65 6E 74 65 72 20 6F
:3920=6E 65 20 6F 72 20 6D 6F
:3928=72 65 20 6F 66 3A 0D 6E
:3930=75 6D 62 65 72 3D 72 65
:3938=70 65 61 74 20 63 6F 75
:3940=6E 74 2C 47 3D 72 65 70
:3948=6C 61 63 65 20 69 6E 20
:3950=65 6E 74 69 72 65 20 66
:3958=69 6C 65 2C 4E 3D 72 65
:3960=70 6C 61 63 65 20 6E 6F
:3968=20 61 73 6B 00 68 5F 53
:3970=50 3C 50 00 00 03 60 11
:3978=E0 38 CD 0B 00 2A 0E 00
:3980=22 23 3A 3E 01 32 72 42
:3988=DD CB 00 96 DD CB 00 9E
:3990=DD CB 00 A6 21 6D 39 0E
:3998=00 18 03 CD BF 38 3E 01
:39A0=CD 1B 00 FE 03 28 4E FE
:39A8=08 28 F0 FE 0D 28 0F FE
:39B0=20 38 EB CD 13 00 77 0C
:39B8=23 3E 09 B9 20 E0 36 00
:39C0=21 6D 39 7E 23 CD 51 14
:39C8=B7 C8 FE 30 38 F5 FE 3A
:39D0=38 33 FE 47 28 23 FE 4E
:39D8=28 25 FE 3F 20 E5 21 00
:39E0=04 11 F8 38 CD 13 18 CD
:39E8=D2 32 2A 23 3A 22 0E 00
:39F0=CD D2 32 18 8E F6 FF 37
:39F8=C9 DD CB 00 D6 18 C4 DD
:3A00=CB 00 DE 18 BE D6 30 32
:3A08=72 42 47 7E 23 FE 30 38
:3A10=B2 FE 3A 30 B0 D6 30 4F
:3A18=78 87 87 80 87 81 32 72
:3A20=42 18 A0 4F 00 DD CB 00
:3A28=7E C8 CD 65 35 DD CB 00
:3A30=6E CA 29 38 C3 66 3A CD
:3A38=3F 41 DD CB 00 EE CD 7D
:3A40=38 DA 70 41 B7 CA 70 41
:3A48=32 7A 42 11 D0 38 CD 0B
:3A50=00 21 B0 42 CD 89 38 DA
:3A58=70 41 32 7B 42 CD 77 39
:3A60=DA 70 41 CD 29 41 DD CB
:3A68=00 B6 3A 72 42 32 73 42
:3A70=DD CB 00 FE DD CB 00 56
:3A78=20 2B CD 65 35 CD E8 3A
:3A80=38 4C CD 05 3B 38 10 DD
:3A88=CB 00 4E 20 13 CD 55 42
:3A90=38 05 DD 35 02 20 E3 CD
:3A98=C5 1D CD 8B 33 C3 98 41
:3AA0=CD A2 33 18 F2 CD 65 35
:3AA8=2A 8E 4A 22 D7 3A 2A 92
:3AB0=4A 22 8E 4A CD E8 3A 38
:3AB8=20 CD 05 3B 38 D9 DD CB
:3AC0=00 4E 20 DC CD 55 42 38
:3AC8=CE CD 65 35 18 E6 DD CB
:3AD0=00 76 CA 53 41 18 C0 52
:3AD8=00 DD CB 00 76 20 B8 2A
:3AE0=D7 3A 22 8E 4A C3 53 41
:3AE8=2A 8E 4A CD 50 38 D8 DD
:3AF0=CB 00 F6 22 8E 4A CD EB
:3AF8=41 38 05 CD 98 41 B7 C9
:3B00=CD 8F 41 B7 C9 DD CB 00
:3B08=5E 20 5B CD DA 32 21 3F
:3B10=00 11 E1 3B CD 13 18 3E
:3B18=17 32 26 00 CD BA 04 3E
:3B20=07 32 26 00 2A 0E 00 22
:3B28=23 3A CD E3 32 3E 01 CD
:3B30=1B 00 CD 51 14 FE 4E 28
:3B38=1C FE 59 28 09 FE 03 20
:3B40=EC CD 55 3B 37 C9 2A 23
:3B48=3A 22 0E 00 CD 13 00 CD
:3B50=E3 32 CD 79 3B CD DA 32
:3B58=21 3F 00 22 0E 00 CD D2
:3B60=32 CD E3 32 B7 C9 CD 79
:3B68=3B B7 C9 2A 8E 4A 3A 7B
:3B70=42 4F 06 00 B7 ED 42 EB
:3B78=C9 3A 7A 42 B7 37 C8 47
:3B80=3A 7B 42 B7 CA CC 3B 4F
:3B88=90 28 23 30 07 78 91 CD
:3B90=CE 3B 18 1A 4F 06 00 ED
:3B98=43 4F 37 CD B4 34 DD CB
:3BA0=00 4E C0 ED 4B 4F 37 2A
:3BA8=8E 4A 09 22 8E 4A CD 6B
:3BB0=3B 21 B0 42 D5 ED B0 E1
:3BB8=22 8E 4A D5 E5 CD E9 35
:3BC0=E1 D1 CD 06 33 ED 53 8E
:3BC8=4A C3 E9 35 48 3E 4F 06
:3BD0=00 ED 43 4F 37 2A 8E 4A
:3BD8=B7 ED 42 22 8E 4A C3 6C
:3BE0=34 52 65 70 6C 61 63 65
:3BE8=20 28 59 2F 4E 29 3F 00
:3BF0=CD 0A 41 3E 5E CD 13 00
:3BF8=3E 40 81 CD 13 00 21 FB
:3C00=40 E5 CD CD 40 CD E3 32
:3C08=FE 42 CA 40 3C FE 4B CA
:3C10=46 3C FE 59 CA 8A 3D FE
:3C18=43 CA E2 3D FE 48 CA 0F
```

```
:3C20=3D FE 56 CA 22 3E FE 53
:3C28=CA FA 3E FE 51 CA C2 3E
:3C30=FE 57 CA 06 3F FE 52 CA
:3C38=07 3F FE 50 CA 6B 40 C9
:3C40=FD 21 E7 42 18 04 FD 21
:3C48=EA 42 CD 65 35 CD C4 3C
:3C50=DA A9 3C ED 5B 8E 4A CD
:3C58=33 42 C8 D2 B6 3C CD E5
:3C60=3C C0 CD DE 3C CD F0 3C
:3C68=2B 22 8E 4A EB CD EB 41
:3C70=DA A5 3C CD 0D 42 CD 1E
:3C78=42 D2 9C 3C E5 D5 CD A5
:3C80=3C D1 E1 ED 53 D7 3A 22
:3C88=8E 4A E5 CD CF 35 E1 CD
:3C90=06 33 CD E3 32 2A D7 3A
:3C98=22 8E 4A C9 2A E2 42 CD
:3CA0=F8 30 C3 CF 35 EB C3 06
:3CA8=3D CD E5 3C C0 2A 8E 4A
:3CB0=CD D7 3C C3 06 3D CD E5
:3CB8=3C C0 CD DE 3C 23 CD F0
:3CC0=3C C3 69 3C ED 5B 92 4A
:3CC8=CD 8B 36 EB FD 7E FF ED
:3CD0=B1 28 02 37 C9 2B B7 FD
:3CD8=75 00 FD 74 01 C9 FD 6E
:3CE0=00 FD 66 01 C9 FD 4E FF
:3CE8=CD 39 42 DD CB 00 4E C9
:3CF0=ED 5B 8E 4A ED 53 D7 3A
:3CF8=22 8E 4A CD 64 34 2A D7
:3D00=3A ED 5B 8E 4A C9 CD DA
:3D08=32 CD 06 33 C3 E3 32 CD
:3D10=65 35 FD 21 E7 42 CD 1D
:3D18=3D FD 21 EA 42 2A 8E 4A
:3D20=7E FD BE FF CA 5F 3D 22
:3D28=D7 3A CD C4 3C D8 22 8E
:3D30=4A CD 64 34 ED 5B D7 3A
:3D38=CD 33 42 C8 30 01 1B ED
:3D40=53 8E 4A CD EB 41 D8 ED
:3D48=53 D7 3A 22 8E 4A E5 CD
:3D50=CF 35 E1 CD 06 33 2A D7
:3D58=3A 22 8E 4A C3 CF 35 CD
:3D60=64 34 CD 06 33 C3 CF 35
:3D68=FD 21 E7 42 CD C4 3C 38
:3D70=15 FD 21 EA 42 CD C4 3C
:3D78=38 0C ED 5B E7 42 CD 33
:3D80=42 38 03 F6 FF C9 F6 FF
:3D88=37 C9 3E 02 32 4E 37 CD
:3D90=68 3D DA 48 41 CD 8E 36
:3D98=ED 43 4F 37 CD 65 35 2A
:3DA0=8E 4A CD A5 3E FE 01 20
:3DA8=06 CD D9 3D C3 95 41 B7
:3DB0=28 19 CD D3 3D ED 5B 4F
:3DB8=37 2A D7 3A B7 ED 52 22
:3DC0=8E 4A CD 7E 3E C2 95 41
:3DC8=C3 CF 35 CD D3 3D 2A D7
:3DD0=3A 18 EC 2A 8E 4A 22 D7
:3DD8=3A 2A E7 42 22 8E 4A C3
:3DE0=6C 34 CD 68 3D DA 48 41
:3DE8=E5 2A 8E 4A CD A5 3E E1
:3DF0=FE 01 C8 CD 8E 36 0B 0B
:3DF8=ED 43 4F 37 CD B4 34 DD
:3E00=CB 00 4E C0 EB 23 3A EC
:3E08=42 FE 02 28 04 FE 01 C8
:3E10=09 ED 5B 8E 4A D5 ED B0
:3E18=E1 CD DA 32 CD 49 33 C3
:3E20=E3 32 3E 02 32 4E 37 CD
:3E28=68 3D DA 48 41 E5 2A 8E
:3E30=4A CD A5 3E E1 FE 01 C8
:3E38=CD 8E 36 ED 43 4F 37 CD
:3E40=B4 34 DD CB 00 4E C0 EB
:3E48=3A EC 42 FE 01 C8 FE 02
:3E50=28 01 09 E5 ED 5B 8E 4A
:3E58=ED 53 D7 3A ED B0 E1 22
:3E60=8E 4A CD 6C 34 2A D7 3A
:3E68=3A EC 42 FE 01 C8 B7 28
:3E70=07 ED 4B 4F 37 B7 ED 42
:3E78=22 8E 4A C3 95 41 2A EA
:3E80=42 CD EB 41 30 19 2A E7
:3E88=42 CD EB 41 30 11 ED 5B
:3E90=54 37 CD 33 42 30 0C 2A
:3E98=EA 42 CD 33 42 38 04 3E
:3EA0=01 B7 C9 AF C9 D5 ED 5B
:3EA8=E7 42 AF CD 33 42 38 0D
:3EB0=3C ED 5B EA 42 CD 33 42
:3EB8=28 03 38 01 3C 32 EC 42
:3EC0=D1 C9 CD 3F 41 21 02 02
:3EC8=11 E1 3E CD 13 18 3E 01
:3ED0=CD 1B 00 CD 13 00 CD 51
:3ED8=14 FE 59 CA 19 2E C3 74
:3EE0=41 0D 41 62 61 6E 64 6F
:3EE8=6E 20 61 20 46 69 6C 65
:3EF0=20 3F 20 28 59 2F 4E 29
:3EF8=20 00 CD 3F 41 CD 03 40
:3F00=DC DF 40 C3 74 41 C9 CD
:3F08=3F 41 CD 13 3F DC DF 40
:3F10=C3 74 41 21 02 02 11 52
:3F18=3F CD 13 18 11 00 FF CD
:3F20=03 00 D8 ED 5B 90 4A 2A
:3F28=94 4A CD 8E 36 EB CD 91
:3F30=3F 38 10 ED 5B 92 4A 2A
:3F38=94 4A CD 8E 36 EB AF ED
:3F40=B1 28 07 2A 90 4A 36 00
:3F48=37 C9 2B 36 00 22 90 4A
:3F50=B7 C9 4C 6F 61 64 20 46
:3F58=69 6C 65 20 4E 61 6D 65
:3F60=20 3F 20 3A 00 0D 4C 6F
:3F68=61 64 20 46 69 6C 65 20
:3F70=53 69 7A 65 20 69 73 20
:3F78=42 69 67 67 65 72 20 74
:3F80=68 61 6E 20 54 65 78 74
:3F88=20 53 69 7A 65 20 21 21
:3F90=00 CD BE 40 E5 11 00 FF
:3F98=CD 8B 13 3E 04 32 80 14
:3FA0=21 00 FF 01 20 00 CD 41
:3FA8=00 38 2A CD 4E 13 28 0D
:3FB0=3E 05 CD EC 0D 11 6A 14
:3FB8=CD 21 13 18 E3 11 62 14
:3FC0=CD 21 13 ED 4B 12 FF CD
:3FC8=EE 3F E1 38 0A CD 44 00
:3FD0=38 05 B7 18 02 E1 37 C3
:3FD8=5D 40 0D 53 61 76 65 20
:3FE0=46 69 6C 65 20 4E 61 6D
:3FE8=65 20 3F 20 3A 00 E5 2A
```

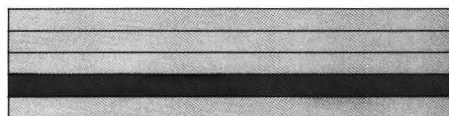
```
:3FF0=58 37 B7 ED 42 E1 D0 F5
:3FF8=D5 11 65 3F CD 0B 00 D1
:4000=F1 37 C9 11 DA 3F 2A 92
:4008=4A 22 74 42 2A 90 4A 22
:4010=76 42 21 02 02 CD 13 18
:4018=11 00 FF CD 03 00 D8 ED
:4020=5B 74 42 2A 76 42 CD 8E
:4028=36 EB CD 2E 40 C9 CD BE
:4030=40 E5 C5 ED 43 92 14 ED
:4038=53 94 14 21 00 00 22 96
:4040=14 11 00 FF CD 8B 13 21
:4048=80 14 36 04 11 5A 14 CD
:4050=21 13 01 20 00 CD 3B 00
:4058=C1 E1 D4 3E 00 F5 3E 01
:4060=CD EC 0D 2A 78 42 22 7E
:4068=14 F1 C9 CD 68 3D DA 48
:4070=41 13 2B 18 07 ED 5B 92
:4078=4A 2A 90 4A CD 8E 36 EB
:4080=CD B7 40 3E 0D CD A1 40
:4088=7E B7 28 0A CD A1 40 D8
:4090=23 0B 78 B1 20 F2 3A BD
:4098=40 B7 C8 3E 0C CD D5 1D
:40A0=C9 FE 0D C2 D5 1D CD D5
:40A8=1D D8 3A BD 40 3C FE 3C
:40B0=20 06 3E 0C CD D5 1D AF
:40B8=32 BD 40 B7 C9 02 E5 2A
:40C0=7E 14 22 78 42 21 DF 40
:40C8=22 7E 14 E1 C9 3E 01 CD
:40D0=1B 00 FE 20 30 02 C6 40
:40D8=CD 13 00 CD 51 14 C9 F5
:40E0=CD F7 07 D5 11 F0 40 CD
:40E8=0B 00 D1 F1 37 C3 5D 40
:40F0=0D 45 72 72 6F 72 20 21
:40F8=21 0D 00 CD 0A 41 CD BA
:4100=04 CD BA 04 CD BA 04 C3
:4108=E3 32 CD DA 32 21 00 00
:4110=22 0E 00 C9 F5 AF 32 1E
:4118=00 3C 32 16 00 3E 4F 32
:4120=1F 00 3E 05 32 17 00 F1
:4128=C9 F5 AF 32 1E 00 3E 06
:4130=32 16 00 3E 4F 32 1F 00
:4138=3E 18 32 17 00 F1 C9 CD
:4140=65 35 CD 14 41 C3 CA 32
:4148=11 C0 41 CD DA 32 CD C2
:4150=32 18 0C CD C2 32 11 7D
:4158=42 CD 0B 00 11 9E 41 CD
:4160=0B 00 CD F7 07 3E 01 CD
:4168=1B 00 FE 0D 20 F7 18 04
:4170=DD CB 00 BE CD 8B 33 CD
:4178=86 41 C3 E3 32 CD FB 32
:4180=32 0F 00 CD 4B 35 2A 54
:4188=37 CD EC 32 C3 49 33 CD
:4190=7D 41 C3 CF 35 CD 4B 35
:4198=CD 86 41 C3 CF 35 0D 20
:41A0=20 2A 2A 20 4E 6F 74 20
:41A8=46 6F 75 6E 64 20 2A 2A
:41B0=20 2C 50 72 65 73 73 20
:41B8=52 65 74 75 72 6E 1A 00
:41C0=20 20 2A 2A 20 4D 61 72
:41C8=6B 65 72 20 53 65 74 20
:41D0=45 72 72 6F 72 20 2A 2A
:41D8=20 2C 50 72 65 73 73 20
:41E0=52 65 74 75 72 6E 1A 00
:41E8=2A 8E 4A D5 ED 5B 54 37
:41F0=CD 33 42 38 13 3A 51 37
:41F8=B7 20 09 ED 5B 56 37 CD
:4200=33 42 30 04 D1 F6 FF C9
:4208=D1 F6 FF 37 C9 D5 E5 CD
:4210=3F 36 22 E2 42 CD 6B 36
:4218=22 E4 42 E1 D1 C9 D5 ED
:4220=5B E2 42 CD 33 42 38 E0
:4228=ED 5B E4 42 CD 33 42 30
:4230=D7 18 D1 E5 B7 ED 52 E1
:4238=C9 C5 D5 E5 F5 21 01 00
:4240=22 4F 37 CD B4 34 DD CB
:4248=00 4E 20 04 2A 8E 4A 71
:4250=F1 E1 D1 C1 C9 F5 3E E6
:4258=CD FE 0D CD 49 0B 32 ED
:4260=42 CD 49 0B 32 EE 42 FE
:4268=03 28 03 F1 B7 C9 F1 37
:4270=C9 00 32 76 F5 F1 FE 20
:4278=C8 32 6A 76 00 CD 99 58
:4280=E3 3A 5C 76 B6 32 5C 76
:4288=23 E3 C9 CD 81 5C CA CE
:4290=58 3A B2 03 CD 99 58 3A
:4298=B9 03 C3 22 59 3A B1 03
:42A0=CD 99 58 C3 19 59 CD 81
:42A8=5C 3E 0C CA 29 57 CD 00
:42B0=53 CD B1 58 20 C3 C2 58
:42B8=3A B4 03 C3 F2 58 3A B3
:42C0=03 CD 99 58 E3 CD D6 4B
:42C8=DA 0F 59 FE 0A CA 18 59
:42D0=FE 8A C2 10 59 CD 81 5C
:42D8=CA 18 59 C3 17 59 23 3A
:42E0=B0 03 CD AE 58 40 1C E3
:42E8=3A 1D 76 B7 CC DB 5B CD
:42F0=DA 32 CD 4B 35 CD EC 32
:42F8=CD CA 32 11 3F 43 CD 29
:4300=43 3E 01 CD 1B 00 FE 0D
:4308=28 13 FE 0A 20 F3 11 80
:4310=45 CD 29 43 3E 01 CD 1B
:4318=00 FE 0D 20 F7 CD EC 32
:4320=2A 54 37 CD 49 33 C3 E3
:4328=32 1A 13 B7 C8 FE 1C 38
:4330=09 FE 20 30 05 CD C8 04
:4338=18 EF CD 13 00 18 EA 09
:4340=09 3C 3C 20 43 75 72 73
:4348=6F 72 20 4D 6F 76 65 6D
:4350=65 6E 74 73 20 3E 3E 0D
:4358=0D 09 09 20 20 20 20 20
:4360=20 20 45 0D 09 09 20 20
:4368=20 20 41 20 53 20 44 20
:4370=46 20 20 6F 72 20 43 75
:4378=72 73 6F 72 20 4B 65 79
:4380=0D 09 09 20 20 20 20 20
:4388=20 20 58 0D 0D 09 5E 53
:4390=2C 1D 09 20 4C 65 66 74
:4398=20 43 68 61 72 61 63 74
:43A0=65 72 09 09 5E 44 2C 1C
:43A8=09 20 52 69 67 68 74 20
:43B0=43 68 61 72 61 63 74 65
:43B8=72 0D 09 5E 41 09 20 4C
```

```
:43C0=65 66 74 20 57 6F 72 64
:43C8=09 09 5E 46 09 20 52 69
:43D0=67 68 74 20 57 6F 72 64
:43D8=0D 09 5E 45 2C 1E 09 20
:43E0=55 70 20 31 20 4C 69 6E
:43E8=65 09 09 5E 58 2C 1F 09
:43F0=20 44 6F 77 6E 20 31 20
:43F8=4C 69 6E 65 0D 09 5E 52
:4400=09 20 46 69 6C 65 20 55
:4408=70 20 53 63 72 65 65 6E
:4410=09 09 5E 43 09 20 46 69
:4418=6C 65 20 44 6F 77 6E 20
:4420=53 63 72 65 65 6E 0D 09
:4428=5E 57 09 20 46 69 6C 65
:4430=20 55 70 20 31 20 4C 69
:4438=6E 65 09 09 5E 5A 09 20
:4440=46 69 6C 65 20 44 6F 77
:4448=6E 20 31 20 4C 69 6E 65
:4450=0D 09 5E 51 52 09 20 54
:4458=6F 20 54 6F 70 20 6F 66
:4460=20 46 69 6C 65 09 09 5E
:4468=51 43 09 20 54 6F 20 45
:4470=6E 64 20 6F 66 20 46 69
:4478=6C 65 0D 09 5E 51 42 09
:4480=20 42 65 67 69 6E 20 4D
:4488=61 72 6B 65 72 09 09 5E
:4490=51 4B 09 20 45 6E 64 20
:4498=4D 61 72 6B 65 72 0D 09
:44A0=5E 49 2C 48 54 41 42 09
:44A8=20 54 41 42 20 53 65 74
:44B0=09 09 5E 48 2C 44 45 4C
:44B8=09 20 4C 65 66 74 20 43
:44C0=68 61 72 61 63 74 65 72
:44C8=0D 0D 09 09 3C 3C 20 49
:44D0=6E 73 65 72 74 20 26 20
:44D8=44 65 6C 65 74 65 20 3E
:44E0=3E 0D 0D 09 5E 56 09 20
:44E8=49 6E 73 65 72 74 20 4D
:44F0=6F 64 65 20 4F 6E 2F 4F
:44F8=66 66 0D 09 5E 47 09 20
:4500=44 65 6C 65 74 65 20 43
:4508=68 61 72 61 63 74 65 72
:4510=20 75 6E 64 65 72 20 43
:4518=75 72 73 6F 72 0D 09 5E
:4520=59 09 20 44 65 6C 65 74
:4528=65 20 4C 69 6E 65 0D 09
:4530=5E 54 09 20 44 65 6C 65
:4538=74 65 20 74 6F 20 45 6E
:4540=64 20 6F 66 20 4C 69 6E
:4548=65 0D 0D 09 28 54 79 70
:4550=65 20 5E 4A 20 46 6F 72
:4558=20 4E 65 78 74 20 46 72
:4560=61 6D 65 20 6F 72 20 52
:4568=65 74 75 72 6E 20 74 6F
:4570=20 4F 72 69 67 69 6E 61
:4578=6C 20 46 69 6C 65 29 00
:4580=0D 0D 09 5E 51 46 09 20
:4588=46 69 6E 64 0D 09 5E 51
:4590=41 09 20 46 69 6E 64 20
:4598=26 20 52 65 70 6C 61 63
:45A0=65 0D 09 5E 4C 09 20 46
:45A8=69 6E 64 2F 52 65 70 6C
:45B0=61 63 65 20 41 67 61 69
:45B8=6E 0D 09 5E 50 09 20 50
:45C0=72 69 6E 74 20 54 65 78
:45C8=74 0D 09 5E 4A 09 20 44
:45D0=69 73 70 6C 61 79 20 48
:45D8=65 6C 70 20 46 69 6C 65
:45E0=0D 09 5E 51 4D 09 20 52
:45E8=65 74 75 72 6E 20 74 6F
:45F0=20 4D 61 69 6E 20 4D 65
:45F8=6E 75 0D 0D 09 09 3C 3C
:4600=20 42 6C 6F 63 6B 20 4F
:4608=70 65 72 61 74 69 6F 6E
:4610=20 3E 3E 0D 0D 09 5E 4B
:4618=42 09 20 42 65 67 69 6E
:4620=20 42 6C 6F 63 6B 09 09
:4628=5E 4B 4B 09 20 45 6E 64
:4630=20 42 6C 6F 63 6B 0D 09
:4638=5E 4B 43 09 20 43 6F 70
:4640=79 20 42 6C 6F 63 6B 09
:4648=09 5E 4B 56 09 20 4D 6F
:4650=76 65 20 42 6C 6F 63 6B
:4658=0D 09 5E 4B 59 09 20 44
:4660=65 6C 65 74 65 20 42 6C
:4668=6F 63 6B 09 09 5E 4B 48
:4670=09 20 48 69 64 65 20 4D
:4678=61 72 6B 65 72 0D 09 5E
:4680=4B 53 09 20 53 61 76 65
:4688=20 54 65 78 74 09 09 5E
:4690=4B 52 09 20 52 65 61 64
:4698=20 54 65 78 74 0D 09 5E
:46A0=4B 50 09 20 50 72 69 6E
:46A8=74 20 42 6C 6F 63 6B 09
:46B0=09 5E 4B 51 09 20 41 62
:46B8=61 6E 64 6F 6E 20 61 20
:46C0=46 69 6C 65 0D 0D 09 28
:46C8=54 79 70 65 20 52 65 74
:46D0=75 72 6E 20 74 6F 20 4F
:46D8=72 69 67 69 6E 61 6C 20
:46E0=46 69 6C 65 29 00 21 D3
:46E8=47 22 19 00 21 06 48 22
:46F0=39 00 3E DF 32 D8 49 32
:46F8=DB 49 21 40 47 22 7E 14
:4700=3E 50 32 06 00 AF 32 72
:4708=14 21 E6 46 22 8C 4A 31
:4710=7E 4A 01 12 47 C5 ED 73
:4718=8A 4A CD A3 04 3E 2A CD
:4720=13 00 CD FD 10 30 FB 1A
:4728=FE 2A C0 13 CD 50 14 13
:4730=D9 21 5E 47 06 0B BE 23
:4738=28 16 23 23 10 F8 D9 C9
:4740=11 56 47 CD A3 04 CD 0B
:4748=00 3E 01 CD EC 0D 18 BF
:4750=5E 23 56 D5 D9 C9 45 72
:4758=72 6F 72 20 21 00 44 8B
:4760=11 4D 1D 12 50 61 10 46
:4768=53 12 52 D0 14 53 6A 10
:4770=4C 9A 10 56 E1 10 54 BE
:4778=12 47 7F 47 58 60 48 1A
:4780=FE 20 2A 8C 4A 28 05 CD
:4788=1F 11 38 27 22 CD 49 CD
```

```
:4790=1F 11 38 19 E5 CD 1F 11
:4798=38 09 22 DC 49 7E 32 DB
:47A0=49 36 DF E1 22 D9 49 7E
:47A8=32 D8 49 36 DF 2A CD 49
:47B0=22 8C 4A F3 31 7E 4A ED
:47B8=5B 8C 4A 2A 8A 4A 2B 72
:47C0=2B 73 22 8A 4A F1 C1 D1
:47C8=E1 DD E1 FD E1 ED 7B 8A
:47D0=4A FB C9 E5 D5 F5 21 06
:47D8=00 39 5E 23 56 1B 3A D8
:47E0=49 FE DF 28 17 2A D9 49
:47E8=B7 ED 52 28 15 3A DB 49
:47F0=FE DF 28 08 2A DC 49 B7
:47F8=ED 52 28 06 F1 D1 E1 C3
:4800=0F 47 F3 F1 D1 E1 F3 ED
:4808=73 8A 4A 31 8A 4A FD E5
:4810=DD E5 E5 D5 C5 F5 2A 8A
:4818=4A 5E 23 56 23 22 8A 4A
:4820=1B ED 53 8C 4A FB 3A D8
:4828=49 FE DF 28 0F 2A D9 49
:4830=77 3A DB 49 FE DF 28 04
:4838=2A DC 49 77 AF 32 72 14
:4840=21 58 48 CD C2 49 2A 8C
:4848=4A 22 CB 49 CD 02 12 CD
:4850=A7 04 CD 06 49 C3 0F 47
:4858=0D 42 72 65 61 6B 20 00
:4860=3A 72 14 F5 AF 32 72 14
:4868=CD 70 48 F1 32 72 14 C9
:4870=1A B7 CA 06 49 CD 51 14
:4878=2E 20 67 13 1A B7 28 08
:4880=CD 51 14 6F 13 1A B7 C0
:4888=EB 01 00 10 21 87 49 7E
:4890=BA 23 20 04 7E BB 28 06
:4898=23 23 0C 10 F2 C9 2B CD
:48A0=A7 04 CD C2 49 3E 3D CD
:48A8=13 00 06 00 79 FE 08 30
:48B0=12 21 7E 4A 09 7E CD 07
:48B8=12 E5 CD EB 48 38 22 7D
:48C0=E1 77 C9 D6 08 87 4F 21
:48C8=7E 4A 09 23 7E CD 07 12
:48D0=2B E5 7E CD 07 12 CD E3
:48D8=48 38 06 EB E1 73 23 72
:48E0=C9 E1 C9 3E 1D CD 13 00
:48E8=CD 13 00 3E 1D CD 13 00
:48F0=CD 13 00 11 00 FF CD 03
:48F8=00 D8 1A B7 37 C8 13 FE
:4900=3D 20 F7 C3 1F 11 06 0E
:4908=CD BC 49 21 7E 49 CD C2
:4910=49 CD 46 14 11 8A 49 CD
:4918=5B 49 2A 7E 4A E5 7C CD
:4920=07 12 06 03 CD BC 49 11
:4928=87 49 CD 5B 49 E1 7D CD
:4930=07 12 3E 28 CD 20 14 06
:4938=08 26 18 29 7C CD 20 14
:4940=10 F7 3E 29 CD 20 14 CD
:4948=46 14 11 A2 49 21 80 4A
:4950=06 03 CD 66 49 06 04 CD
:4958=66 49 C9 EB CD C2 49 EB
:4960=13 3E 3D C3 20 14 CD 5B
:4968=49 D5 5E 23 56 23 EB CD
:4970=02 12 EB D1 10 03 C3 46
:4978=14 CD B7 49 18 E8 53 5A
:4980=20 48 20 50 4E 43 00 46
:4988=20 00 41 20 00 43 20 00
:4990=42 20 00 45 20 00 44 20
:4998=00 4C 20 00 48 20 00 41
:49A0=46 00 42 43 00 44 45 00
:49A8=48 4C 00 49 58 00 49 59
:49B0=00 53 50 00 50 43 00 3E
:49B8=20 C3 20 14 CD B7 49 10
:49C0=FB C9 7E B7 C8 CD 20 14
:49C8=23 18 F7 60 CD A0 14 C1
:49D0=F1 C9 F5 0A E6 7E 02 F1
:49D8=DF F5 C5 DF 01 26 77 E5
:49E0=21 83 43 86 E1 C3 29 60
:49E8=3A 82 43 F5 C5 D5 CD 68
:49F0=60 E5 21 87 43 96 E1 2F
:49F8=16 7F C3 38 60 F5 C5 D5
:4A00=57 3A 81 43 01 26 77 5F
:4A08=1C 1D CA 48 60 0A A2 02
:4A10=03 03 03 03 C3 3A 60 D1
:4A18=C1 F1 C9 C5 47 CD 1B 31
:4A20=4F 78 B9 C1 C9 E5 CD 1B
:4A28=31 21 B1 36 96 2F 3C E1
:4A30=C9 CD 1B 31 32 1E 76 F5
:4A38=E5 6F 17 9F 67 29 29 01
:4A40=42 77 09 44 4D E1 F1 C9
:4A48=4D 0B 81 35 55 03 4D 0B
:4A50=4D 0B 08 08 12 08 80 27
:4A58=80 27 50 01 A0 0B A8 06
:4A60=4D 0B 01 1A 55 03 FD FF
:4A68=9F 10 09 30 6B 4A 08 20
:4A70=00 4A 9F 4A 08 04 C4 11
:4A78=78 4A A6 11 12 47 00 00
:4A80=00 00 00 00 00 00 00 00
:4A88=00 00 7C 4A E6 46 FE 20
:4A90=96 4A 96 4A FF FD 00 00
:4A98=00 00 00 00 00 00 00 00
```

用語解説

❶ワードスター	ワードスターはマイクロプロ社(米)の英文ワード・プロセッサで，すぐれたエディタとしてCP/Mなどのユーザーに広く使われている。
❷オンメモリ	外部記憶装置を使わず，メモリ(RAM)だけで処理すること。
❸ソース・プログラム	原始プログラムと訳されるが，これはアセンブルされる前のプログラムやコンパイルされる前のプログラムのこと。

第4章

マシン語
プログラミングの
定石と実践テクニック

4-1 プロローグ
4-2 定石
4-3 実践テクニック

4-1 プロローグ

1章で基礎知識，2章でマシン語命令を紹介してきましたが，専門用語と命令が多過ぎるために途方にくれている方が多いのではないでしょうか。

コンピュータの専門用語にはたいへんな数があり，また抽象的なものも多いのでなかなか把握しにくいものです。しかし，本書で扱っているくらいの用語であれば，使っているうちに覚えてしまうでしょう。要するに慣れの問題です。

2章で解説したマシン語命令は，実は半分くらい覚えていれば実用としては困らないのです。フラグについても，プログラムでよく使うのはCYフラグとZフラグくらいでP/VフラグやSフラグは，当面ほとんど使うことはないでしょう。また，NフラグとHフラグはCPU内部で使われるだけで全く無視してしまってもかまいません。ただし，2章は後で資料として使えるように配慮したものですから，本書を読み終えた後，もう1度2章を眺めてみてください。

さて，本章ではマシン語プログラミングの定石と実践テクニックを取り上げていきますが，定石については2章で触れたものも少しあります。重複になりますが，ここでこれらをまとめてみましょう。

1 Aレジスタの内容を0にする

〝LD A, 0〟と〝XOR A〟は両方ともAレジスタの内容を0にする命令です。しかし，前者はマシン・コードにすると2バイトになるのに対し，後者は1バイトですみます。

2 キャリーフラグのリセット

16ビットの減算命令にはSBC命令しかなく，キャリー

を含まない減算をしたい場合，前もってキャリーフラグをリセットしなければなりません。そこで，

```
AND  A  または  OR  A
SBC  HL, DE
```

あるいは，

```
SCF
CCF
SBC  HL, DE
```

とします。この場合も前者の方がバイト数が少ないためよく使われます。

③16ビット・レジスタどうしのロード命令

16ビット・レジスタどうしのロード命令は，ＳＰをデスティネーションとした次の３種類しかありません。

```
LD  SP, HL
LD  SP, IX
LD  SP, IY
```

たとえば，ＨＬレジスタにＤＥレジスタの内容を入れたい場合，

```
LD   H, D
LD   L, E
```

または，

```
PUSH  DE
POP   HL
```

とします。両者ともバイト数は同じですが，前者の方がより高速です。

このほかにも定石と実践テクニックを紹介していきますが，これらはたとえて言えばパズルみたいなものですから，推理小説と同じ感覚で読んでください。

4-2 定石

ここで述べる定石テクニックには，プログラムで比較的よく使われるものとそうでないものがあります。いずれにしろ個々の命令を，より深く理解する手立てになると思われます。

4-2-1 レジスタの内容を交換する

レジスタの内容を交換する命令は，〝EX　DE，HL〟という命令しかありません。そこでまず，８ビットのレジスタＡとＢの内容を交換するプログラムを考えてみます。いろいろと考えられるのですが，

```
LD      H, A
LD      A, B
LD      B, H
```

というようにＨレジスタをワークとして使ってできます。16ビット・レジスタＨＬとＢＣの交換はスタックを使って，

```
PUSH    HL
PUSH    BC
POP     HL
POP     BC
```

として実現できます。

4-2-2 大小比較

BASICではIF文を使って簡単に大小比較ができますが，マシン語の場合は慣れていないと一見しただけでは，何をするプログラムかわかりません。

まず，8ビットで大小比較（符号なし）を行ってみましょう。ここではＣＰ命令と条件付きＪＰ命令を使います。例としてＡレジスタの内容と数値の10を比較してみます。

①A＝10のとき，MAINへジャンプ

```
CP    10
JP    Z, MAIN
```

②A≒10のとき，MAINへジャンプ

```
CP    10
JP    NZ, MAIN
```

③A＜10のとき，MAINへジャンプ

```
CP    10
JP    C, MAIN
```

④A≧10のとき，MAINへジャンプ

```
CP    10
JP    NC, MAIN
```

⑤A≦10のとき，MAINへジャンプするというプログラムはこれまでのように１個のＪＰ命令ではできません。これはＡ＜10(③)またはＡ＝10(①)と考えられますから，

```
CP    10
JP    C, MAIN
JP    Z, MAIN
```

とできます。

⑥A＞10のとき， MAINへジャンプするというプログラムは，Ａ≧10(**④**)でＡ≠10(**②**)と考えられるので，

```
        CP      10
        JP      NC, NEXT
                  :
NEXT:JP         NZ, MAIN
```

としてもできますし，**⑤**でない場合と考えると，

```
        CP      10
        JP      Z, NEXT
        JP      C, NEXT
        JP      MAIN
NEXT:             :
```

とも書けます。

次に16ビットの大小比較を行ってみましょう。例としてＤＥレジスタの内容とＢＣレジスタの内容を比較してみましょう。16ビットのＣＰ命令がないので，SBC命令(16ビット)を使います。この場合，ＨＬレジスタだけしか16ビットのアキュムレータとして使えないので少々面倒になります。

⑦DE＝BCのとき， MAINへジャンプ

```
        EX      DE, HL
        OR      A       ;キャリーフラグをリセット
        SBC     HL, BC
        JP      Z, MAIN
```

⑧DE≠BCのとき， MAINへジャンプ

```
        EX      DE, HL
        OR      A
        SBC     HL, BC
```

```
        JP      NZ, MAIN
```

⑨DE＜BCのとき，MAINへジャンプ

```
        EX      DE, HL
        OR      A
        SBC     HL, BC
        JP      C, MAIN
```

⑩DE≧BCのとき，MAINへジャンプ

```
        EX      DE, HL
        OR      A
        SBC     HL, BC
        JP      NC, MAIN
```

⑪DE≦BCのとき，MAINへジャンプというプログラムは⑤と同様に，

```
        EX      DE, HL
        OR      A
        SBC     HL, BC
        JP      C, MAIN
        JP      Z, MAIN
```

と書けます。DE＞BCのとき，MAINへジャンプするというプログラムは，皆さんが考えてみてください。SBC命令を使う以外に，ペア・レジスタを8ビットに分割して考えることもできます。

⑫DE＝BCのとき，MAINへジャンプするというプログラムは，D＝BかつE＝CならDE＝BCなので，

```
        LD      A, D
        CP      B
        JP      Z, NEXT
             ⋮
```

```
NEXT:LD    A, E
     CP    C
     JP    Z, MAIN
```

と書けます。このほかにも大小比較も考えてみてください。

4-2-3 フィル・メモリ

フィル・メモリとはメモリの特定のエリアを特定のデータで埋めることを指します。実は，これが第1章で紹介したプログラム（**リスト1-2**）のマシン語部分なのです。このソース・プログラムは，

```
LD    HL,0C000H
LD    DE,0C001H
LD    BC,0FFFH
LD    (HL), 1
LDIR
RET
```

となります。これはLDIR命令を使ったちょっとおもしろい例です。この動作を説明しましょう。

1. LD (HL),1でC000番地に1が書き込まれる。

2. LDIRで，まずC000番地の内容がC001番地に入る。
   つまり，C001番地に1が書き込まれる。次にHLがインクリメントされて，HLはC001番地を指し，DEもインクリメントされC002番地を指す。BCはデクリメントされる。すでにC001番地の内容が1になっているのでC002番地にも1が書き込まれる。以上の動作をBCが0になるまで繰り返す。
   というわけで，C000番地からCFFF番地までを1で埋め

つくすことができるわけです。

4-2-4 ループ

BASICでもFOR～NEXTなどのようなループ命令はよく使われます。ここでは，基本的なループを紹介しましょう。まず，Cレジスタをループ・カウンタとして使ってみます。

```
        LD   C, 100
LOOP:        :           } この間の処理を100回
             :           } 繰り返す
        DEC  C
        JR   NZ, LOOP
```

8ビットのレジスタなので，ループできる最大の回数は，256回でこれは次のように書けます。

```
        LD   C, 0
LOOP:        :
             :
        DEC  C
        JR   NZ, LOOP
```

ループ・カウンタとしてCレジスタに入れる値が0であることに注意してください。8ビットの最大値はFFHだからといって，これを入れるとループ回数は255(FFH)になってしまいます。

ループ・カウンタとして8ビット・レジスタを使う場合，Bレジスタだけは特別で，DJNZ命令が使えます。

```
        LD   B, 100
LOOP:        :
             :
        DJNZ LOOP
```

DJNZ命令を使うとバイト数も短くなりますし，Bレジスタを使ったループだとすぐわかるという利点もあります。

ループ回数が多くて8ビットでは足りない場合，16ビット・レジスタを使いますが，

```
        LD   BC, 1000
LOOP:        :
             :
        DEC  BC
        JR   NZ, LOOP
```

というようなプログラムを書くと，間違いなく期待どおり動いてくれません。なぜならば，「ペア・レジスタの内容をデクリメントしてもフラグは一切変化しない」という重大な事項を忘れているからです。この場合，

```
        LD     BC, 1000
LOOP:          :
               :
        DEC    BC
        LD     A, B  } Bレジスタと Cレジスタの
        OR     C     } ORを取る
        JR     NZ, LOOP
```

とするとうまくいきます。

4-2-5 F,SP,PCの値を得る

これらはプログラムで使われることはほとんどないと思いますが，デバッグのときに必要になる場合があります。

1F(フラグ)レジスタの内容を得る

これには，たとえば

```
        PUSH  AF
        POP   BC
```

とするとCレジスタにFレジスタの内容が入るので，あとはBIT命令などを使って値を調べることができます。

2SP(スタック・ポインタ)の値を得る

SPの値を16ビット・レジスタに直接ロードする命令はありません。そこで，

```
        LD    HL, 0
        ADD   HL, SP
```

というようにHLを0にしてから，〝ADD HL, SP″を実行すれば，HLにSPの値が入ります。

3PC(プログラム・カウンタ)の値を得る

PCの値を得るのは少々面倒です。というのは，直接PCの値を扱う命令がないからです。CALL命令だけが次に実行する番地を自動的にスタックに積むので，これを利用します。

```
        CALL  GETPC
        DEC   BC
        DEC   BC
        DEC   BC
        RET
GETPC : POP   BC
        PUSH  BC
        RET
```

〝CALL GETPC″を実行すると，スタックには次の命令（ここでは最初の〝DEC BC″）のマシン・コードのアドレスが入ります。〝CALL GETPC″は3バイトの命令なので3回デクリメントしてやるとCALL命令のアドレスになります。

4-3 実践テクニック

ここでは，実践テクニックとして実際に使えるサブルーチンや実行結果が目に見えるもの（または聴こえるもの）を取り上げます。リストはアセンブル・リストで掲載するので，打ち込んで試してみてください。

4-3-1 パラメータの渡し方

メイン・ルーチンとサブルーチンとの間で，パラメータを受け渡す方法は，いくつか考えられますが，処理する内容によって最適なものを選んでください。

1 レジスタを使う

リスト4-1は，単純な例なのですぐ処理内容はわかると思います。ADDABCというラベルのついたサブルーチンは，A，B，Cのレジスタの内容を加算して結果をAレジスタに入れます。パラメータがこのように少ない場合，プログラムが短くなるので有効でしょう。

リスト4-1

```
X1 Self Assembler  Rev 1.0    PAGE     1


 1:                      ;------------------------------------------
 2:                      ;        LIST     4-1
 3:                      ;
 4:    A000                       ORG      0A000H
 5:    A000 3E01          MAIN:   LD       A,1
 6:    A002 0602                  LD       B,2
 7:    A004 0E03                  LD       C,3
 8:    A006 CD0AA0                CALL     ADDABC
 9:    A009 C9                    RET
10:                      ;
11:    A00A 80           ADDABC:  ADD      A,B
12:    A00B 81                    ADD      A,C
13:    A00C C9                    RET
14:
15:    A00D                       END
```

②アドレス渡し

IOCSの1文字表示("ACCPRT")を使って，文字列を表示するプログラムをつくってみます(IOCSは第5章で取り上げます)。ACCRRTというルーチンは，Aレジスタに入っている数値をASCIIコードと見なして画面に表示するものです。

リスト4-2のメインでは，HLに表示する文字列(MSG1とMSG2)の先頭アドレスを入れて(アドレス渡し)，LINOUTというサブルーチンをコールします。LINOUTでは，0のデータが来るまで1文字表示を続けます。

リスト4-2

```
 X1 Self Assembler  Rev 1.0    PAGE    1

  1:                      ;------------------------------------------
  2:                      ;        LIST    4-2
  3:                      ;
  4:    0013 =            ACCPRT   EQU     0013H
  5:    A000                       ORG     0A000H
  6:    A000 210DA0       MAIN:    LD      HL,MSG1
  7:    A003 CD31A0                CALL    LINOUT
  8:    A006 2120A0                LD      HL,MSG2
  9:    A009 CD31A0                CALL    LINOUT
 10:    A00C C9                    RET
 11:                      ;
 12:    A00D 4920616D MSG1:        DEFM    'I am a computer.'
 13:    A01D 0D0A                  DEFB    0DH,0AH           ; CR & LF
 14:    A01F 00                    DEFB    0                 ; End Marker
 15:    A020 596F7520 MSG2:        DEFM    'You are a man.'
 16:    A02E 0D0A00                DEFB    0DH,0AH,0
 17:                      ;
 18:    A031 7E           LINOUT:  LD      A,(HL)
 19:    A032 B7                    OR      A                 ; End ?
 20:    A033 C8                    RET     Z                 ; Yes,then Return
 21:    A034 CD1300                CALL    ACCPRT
 22:    A037 23                    INC     HL
 23:    A038 18F7                  JR      LINOUT
 24:
 25:    A03A                       END
```

③インライン・パラメータ

これは，レジスタを使わずにサブルーチンにアドレスを渡すものです。**リスト4-3**でCALL命令が実行されるとスタックには戻り番地としてPSGDATのアドレスが入ります。そこで，PSGPLYというサブルーチンでは，最初に"POP　HL"を実行してPSGDATのアドレスを得ています。サブルーチンから戻るときは，HLがすでにA018Hになっているので"JP　(HL)"を使います。RET

命令で戻らないように注意してください。このように，CALL命令のすぐ後に置いたパラメータをインライン・パラメータと呼びます。

リスト4-3の30～39行を**リスト4-4**のように書き換えても結果は同じです。こちらの方がRET命令で戻っているので気分的にすっきりします。

SOUNDというサブルーチンは，BASICのSOUND命令のような働きをするものですが，詳しくは第5章で解説します。

リスト4-3

```
X1 Self Assembler  Rev 1.0    PAGE    1

 1:                     ;----------------------------------------
 2:                     ;         LIST     4-3
 3:                     ;
 4:    013C =           PSGINT  EQU     013CH           ; Initialize PSG
 5:    A000                     ORG     0A000H
 6:    A000 CD29A0      MAIN:   CALL    PSGPLY
 7:    A003 00D9        PSGDAT: DEFB    0,217
 8:    A005 0100                DEFB    1,0
 9:    A007 02C6                DEFB    2,198
10:    A009 0300                DEFB    3,0
11:    A00B 07FC                DEFB    7,252
12:    A00D 0810                DEFB    8,16
13:    A00F 0910                DEFB    9,16
14:    A011 0B81                DEFB    11,129
15:    A013 0C0D                DEFB    12,13
16:    A015 0D08                DEFB    13,8
17:    A017 FF                  DEFB    0FFH            ; End Marker
18:                     ;
19:    A018 1E0A                LD      E,10
20:    A01A 010000      DELAY:  LD      BC,0
21:    A01D 0B          DELAY1: DEC     BC
22:    A01E 78                  LD      A,B
23:    A01F B1                  OR      C
24:    A020 20FB                JR      NZ,DELAY1
25:    A022 1D                  DEC     E
26:    A023 20F5                JR      NZ,DELAY
27:    A025 CD3C01              CALL    PSGINT
28:    A028 C9                  RET
29:                     ;
30:    A029 E1          PSGPLY: POP     HL              ; HL <= PSGDAT
31:    A02A 7E          PLAY:   LD      A,(HL)
32:    A02B 23                  INC     HL
33:    A02C FEFF                CP      0FFH            ; End ?
34:    A02E 2807                JR      Z,EXIT
35:    A030 56                  LD      D,(HL)
36:    A031 CD38A0              CALL    SOUND
37:    A034 23                  INC     HL
38:    A035 18F3                JR      PLAY
39:    A037 E9          EXIT:   JP      (HL)            ; Return
40:                     ;
41:                     ;       SOUND   A,D
42:                     ;
43:    1C00 =           PSGCOM  EQU     1C00H
44:                     ;
45:    A038 C5          SOUND:  PUSH    BC
46:    A039 01001C              LD      BC,PSGCOM
47:    A03C ED79                OUT     (C),A
```

```
 48:     A03E 05                DEC     B
 49:     A03F ED51              OUT     (C),D
 50:     A041 C1                POP     BC
 51:     A042 C9                RET
 52:
 53:     A043                   END
```

リスト4-4

```
 30:     A029 E3      PSGPLY: EX      (SP),HL          ; HL <= PSGDAT
 31:     A02A 7E      PLAY:   LD      A,(HL)
 32:     A02B 23              INC     HL
 33:     A02C FEFF            CP      0FFH             ; End ?
 34:     A02E 2807            JR      Z,EXIT
 35:     A030 56              LD      D,(HL)
 36:     A031 CD39A0          CALL    SOUND
 37:     A034 23              INC     HL
 38:     A035 18F3            JR      PLAY
 39:     A037 E3      EXIT:   EX      (SP),HL
 40:     A038 C9              RET
```

4-3-2 ジャンプ・テーブル

BASICの"ON　変数　GOTO"文をマシン語でつくってみましょう。これは，変数が０ならプログラム０を，１ならプログラム１を…というように変数によって飛び先を変えるものです。これをマシン語で実現するためには，ジャンプ・テーブルと呼ばれるものが必要になります。**リスト4-5**のJPTBLというラベルが付いた6バイトのデータがジャンプ・テーブルです。

リスト4-5では，Ｃレジスタが変数の役目を果すもので，10～16行目でどこへジャンプするか計算します。このONGOTOというルーチンで，初めに"INC　C"を実行しているのは，Cレジスタが0の場合を考えたものです。リストでは，Ｃレジスタに１を入れているので，SUB1というルーチンを実行してメインに戻ってきます。

リスト4-5

```
X1 Self Assembler  Rev 1.0    PAGE    1

  1:                          ;---------------------------------------------------
  2:                          ;       LIST    4-5
  3:                          ;
  4:     A000                         ORG     0A000H
  5:     A000 0E01            MAIN:   LD      C,1
  6:     A002 CD06A0                  CALL    ONGOTO
  7:     A005 C9                      RET
  8:                          ;
```

```
 9:                       ;
10:   A006 DD211CA0 ONGOTO: LD      IX,JPTBL
11:   A00A 110200          LD      DE,2
12:   A00D 0C              INC     C
13:   A00E 0D       ADDJP:  DEC     C
14:   A00F 2804            JR      Z,JUMP
15:   A011 DD19            ADD     IX,DE
16:   A013 18F9            JR      ADDJP
17:                       ;
18:   A015 DD6E00   JUMP:   LD      L,(IX)
19:   A018 DD6601          LD      H,(IX+1)
20:   A01B E9              JP      (HL)
21:                       ;
22:   A01C 22A0     JPTBL:  DEFW    SUB0
23:   A01E 24A0            DEFW    SUB1
24:   A020 27A0            DEFW    SUB2
25:
26:                       ;
27:   A022 AF       SUB0:   XOR     A
28:   A023 C9              RET
29:   A024 3E01     SUB1:   LD      A,1
30:   A026 C9              RET
31:   A027 3E02     SUB2:   LD      A,2
32:   A029 C9              RET
33:
34:   A02A                 END
```

4-3-3 文字列サーチ

文字列をサーチするプログラムは，アドベンチャーゲームなどで入力されたコマンドを処理したり，アセンブラなどを自作しようとするときに必要になります。ここでは，HLレジスタが示すアドレスからの文字列が〝LIST〟なら0，〝RUN〟なら1，〝LOAD〟なら2，その他ならFFHをAレジスタに入れて戻るルーチンをつくってみました(**リスト4-6**)。

データは，

文字列，0，値

という形で並べ，データの最後は，

0，0FFH

とします。

SEARCHというサブルーチンでは，DEレジスタでテーブルを示し，その値が0なら文字列が一致したとみなします。0でないときは，HLレジスタの示す番地の内容と比較し，これが等しくなくなるまで繰り返します。

等しくなければ，テーブルの次の文字列と比較します。このとき，HLレジスタの値をもとに戻さなくてはなりません。

テーブルの最後は0，FFHですから，HLレジスタの示す文字列とそれ以上比較することなく，値をFFHとしてAレジスタに入れます。

例としてここでは，〝RUN〟という文字列をサーチしますから，Aレジスタの値は1になります。

リスト4-6

```
X1 Self Assembler  Rev 1.0    PAGE    1

  1:                      ;-----------------------------------------------
  2:                      ;        LIST    4-6
  3:                      ;
  4:   A000               ORG     0A000H
  5:   A000 2107A0   MAIN:   LD      HL,SRCHDT
  6:   A003 CD0AA0           CALL    SEARCH
  7:   A006 C9               RET
  8:                      ;
  9:   A007 52554E   SRCHDT: DEFM    'RUN'
 10:                      ;
 11:                      ;
 12:   A00A 1123A0   SEARCH: LD      DE,STRTBL
 13:   A00D E5       LOOP:   PUSH    HL
 14:   A00E 1A       LOOP1:  LD      A,(DE)
 15:   A00F 13               INC     DE
 16:   A010 B7               OR      A
 17:   A011 280D             JR      Z,FIND
 18:   A013 BE               CP      (HL)
 19:   A014 23               INC     HL
 20:   A015 28F7             JR      Z,LOOP1
 21:   A017 E1               POP     HL
 22:   A018 1A       NEXT:   LD      A,(DE)
 23:   A019 13               INC     DE
 24:   A01A B7               OR      A
 25:   A01B 20FB             JR      NZ,NEXT
 26:   A01D 13               INC     DE
 27:   A01E 18ED             JR      LOOP
 28:   A020 E1       FIND:   POP     HL
 29:   A021 1A               LD      A,(DE)
 30:   A022 C9               RET
 31:                      ;
 32:   A023 4C495354 STRTBL: DEFM    'LIST'
 33:   A027 0000             DEFB    0,0
 34:   A029 52554E           DEFM    'RUN'
 35:   A02C 0001             DEFB    0,1
 36:   A02E 4C4F4144         DEFM    'LOAD'
 37:   A032 0002             DEFB    0,2
 38:   A034 00FF             DEFB    0,0FFH
 39:
 40:   A036                  END
```

4-3-4 乗除算

Z80には，かけ算や割り算の命令がないので，そのためのサブルーチンをつくらなくてはなりません。ここでは符号なし整数とみなして，乗除算を行いますが，それでも後半部は難しいかも知れません。必要なときは，使い方だけ憶えて使ってください。

1 簡単なかけ算，割り算

リスト4-7のMULT16というサブルーチンは，足し算の繰り返しでかけ算を行っています。

ＤＥレジスタに被乗数，ＢＣレジスタに乗数を入れてこのサブルーチンをコールすると，ＢＣ回だけＤＥレジスタをＨＬレジスタに足します。ループの先頭で終了の判定をしていますから，ＢＣレジスタが０のときは一度も〝ADD HL, DE〞を実行しません。

このサブルーチンを使うときは，ＢＣ，ＤＥ，ＨＬの各レジスタは０～65535までの数を表わすものとします。

かけ算の答えが65535を超えると正しい結果を得ることができません。

リスト4-8のDIV16は，割り算のサブルーチンです。

リスト4-7

```
X1 Self Assembler  Rev 1.0    PAGE    1

 1:                    ;-----------------------------------------------------
 2:                    ;       LIST     4-7
 3:                    ;
 4:   A000                     ORG      0A000H
 5:   A000 116400      MAIN:   LD       DE,100
 6:   A003 016400              LD       BC,100
 7:   A006 CD0AA0              CALL     MULT16
 8:   A009 C9                  RET
 9:                    ;
10:                    ;       HL = DE * BC
11:                    ;
12:   A00A 210000      MULT16: LD       HL,0
13:   A00D 78          LOOP:   LD       A,B
14:   A00E B1                  OR       C
15:   A00F C8                  RET      Z               ; If BC=0 Then Return
16:   A010 19                  ADD      HL,DE
17:   A011 0B                  DEC      BC
18:   A012 18F9                JR       LOOP
19:
20:   A014                     END
```

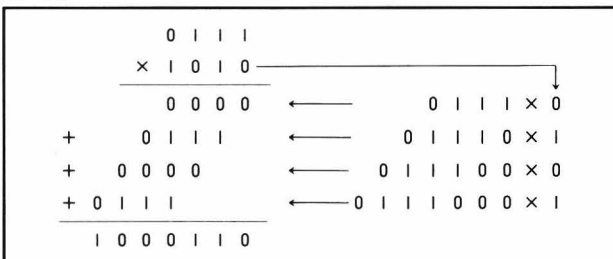
このサブルーチンでは被除数（HLレジスタ）から除数(DEレジスタ)を何回引くことができるかを数えることにより答を求めています。サブルーチン内でBCレジスタに－1を入れているのは，次のループの先頭で〝INC BC〟を実行するためです。このループを実行する前に〝OR　E〟でDEレジスタが0かどうかを調べるとともにキャリーフラグをリセットしています。

リスト4-8

```
X1 Self Assembler  Rev 1.0    PAGE    1

  1:                      ;-------------------------------------------------
  2:                      ;         LIST    4-8
  3:                      ;
  4:     A000                       ORG     0A000H
  5:     A000 211027                LD      HL,10000
  6:     A003 11E803                LD      DE,1000
  7:     A006 CD0AA0                CALL    DIV16
  8:     A009 C9                    RET
  9:                      ;
 10:                      ;         BC = HL / DE
 11:                      ;
 12:     A00A 7A          DIV16:    LD      A,D
 13:     A00B B3                    OR      E                  ; DE=0? & CY Flag Reset
 14:     A00C C8                    RET     Z                  ; If DE=0 Then Return
 15:     A00D 01FFFF                LD      BC,-1
 16:     A010 03          LOOP:     INC     BC
 17:     A011 ED52                  SBC     HL,DE
 18:     A013 30FB                  JR      NC,LOOP
 19:     A015 C9                    RET
 20:
 21:     A016                       END
```

2効率のよいかけ算

リスト4-7のようなかけ算のルーチンでは，最大で65535回もループを回ることになります。これでは，いかにマシン語が高速といっても頻繁に使うと速度が低下してしまいます。**図4-1**のように2進数の筆算を行うようにす

図4-1

れば，もっと効率のよいプログラムを組むことができます。

プログラムは**リスト4-9**のようになります。Bレジスタに被乗数，Cレジスタに乗数を入れてMULT8をコールします。このサブルーチンの〝INC　C″，〝DEC　C″はCレジスタが0かどうかを調べるものです。

〝SRL　C″でCレジスタの最下位ビットをキャリーフラグへ送り，キャリーフラグが1になっていればAレジ

リスト4-9

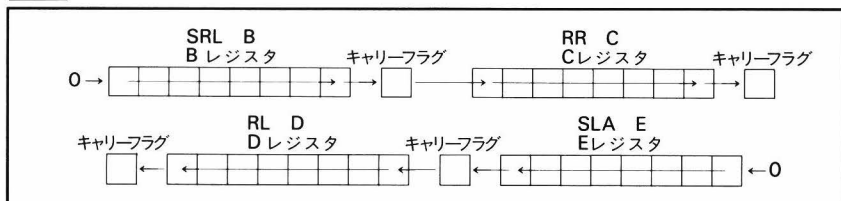
```
X1 Self Assembler  Rev 1.0    PAGE    1

 1:                          ;----------------------------------------------------
 2:                          ;       LIST    4-9
 3:                          ;
 4:     A000                         ORG     0A000H
 5:     A000 060A    MAIN:   LD      B,10
 6:     A002 0E0A                    LD      C,10
 7:     A004 CD08A0                  CALL    MULT8
 8:     A007 C9                      RET
 9:                          ;
10:                          ;       A = B * C
11:                          ;
12:     A008 AF      MULT8:  XOR     A
13:     A009 0C      LOOP:   INC     C
14:     A00A 0D                      DEC     C
15:     A00B C8                      RET     Z
16:     A00C CB39                    SRL     C
17:     A00E 3001                    JR      NC,SKIP
18:     A010 80                      ADD     A,B
19:     A011 CB20    SKIP:   SLA     B
20:     A013 18F4                    JR      LOOP
21:
22:     A015                         END
```

リスト4-10

```
X1 Self Assembler  Rev 1.0    PAGE    1

 1:                          ;--------------------------------------------------
 2:                          ;       LIST    4-10
 3:                          ;
 4:     A000                         ORG     0A000H
 5:     A000 016400  MAIN:   LD      BC,100
 6:     A003 116400                  LD      DE,100
 7:     A006 CD0AA0                  CALL    MULT16
 8:     A009 C9                      RET
 9:                          ;
10:                          ;       HL = BC * DE
11:                          ;
12:     A00A 210000  MULT16: LD      HL,0
13:     A00D 79      LOOP:   LD      A,C
14:     A00E B0                      OR      B
15:     A00F C8                      RET     Z
16:     A010 CB38                    SRL     B
17:     A012 CB19                    RR      C
18:     A014 3001                    JR      NC,SKIP
19:     A016 19                      ADD     HL,DE
20:     A017 CB23    SKIP:   SLA     E
21:     A019 CB12                    RL      D
22:     A01B 18F0                    JR      LOOP
23:
24:     A01D                         END
```

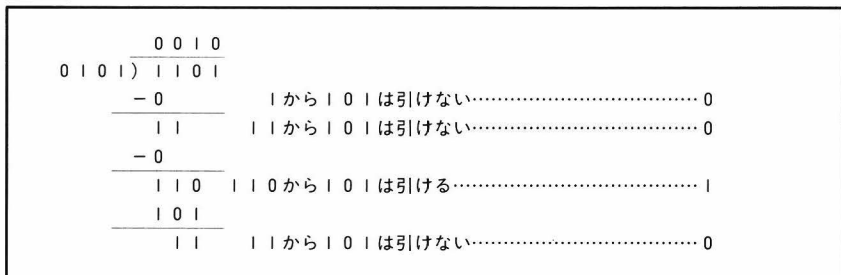
スタにBレジスタを加えます。次に〝SLA　B″で被乗数(Bレジスタ)を2倍にします。このループをCレジスタが0になるまで繰り返します。

同様に16×16ビットの計算を考えてみましょう。**リスト4-10**のMULT16がそのサブルーチンです。ここではBCレジスタの最下位ビットを〝SRL　B″と〝RR　C″でキャリーフラグに送り(**図4-2**)，キャリーフラグが1になればHLレジスタにDEレジスタを加えます。DEレジスタを2倍するには，〝SLA　E″，〝RL　D″を使います(**図4-2**)。

図4-2

3 効率のよい割り算

割り算も筆算の方法を使うと効率のよいプログラムがつくれます。**図4-3**は，2進数の割り算を筆算で計算したものです。

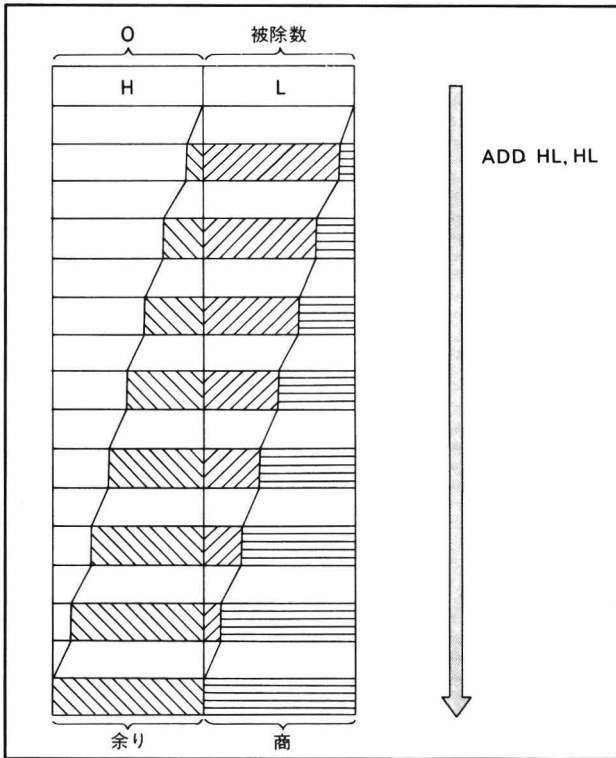
被除数の最上位の桁数を引くことができれば，答の最上位を1に，引くことができなければ答の最上位を0にします。引いた残りを余りとします。これを8回繰り返すと8ビット÷8ビットの割り算ができます。

図4-3

リスト4-11のDIV8というサブルーチンを見てください。初めHレジスタに0を，Bレジスタにループ回数を入れています。

リスト4-11

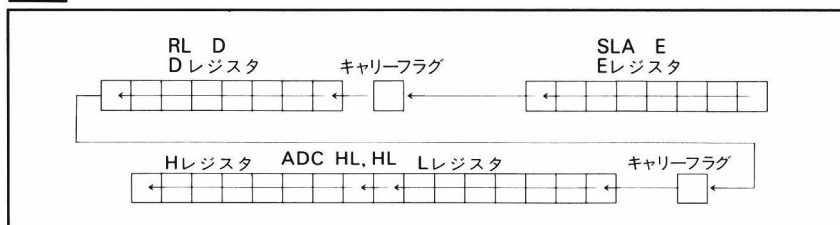
```
 X1 Self Assembler  Rev 1.0    PAGE    1

  1:                       ;-----------------------------------------------
  2:                       ;         LIST    4-11
  3:                       ;
  4:   A000                          ORG     0A000H
  5:   A000 2E64           MAIN:     LD      L,100
  6:   A002 0E0A                     LD      C,10
  7:   A004 CD08A0                   CALL    DIV8
  8:   A007 C9                       RET
  9:                       ;
 10:                       ;         L = L / C
 11:                       ;         H = L MOD C
 12:                       ;
 13:   A008 2600           DIV8:     LD      H,0
 14:   A00A 0608                     LD      B,8            ; Set loop counter
 15:   A00C 29             LOOP:     ADD     HL,HL
 16:   A00D 7C                       LD      A,H
 17:   A00E 91                       SUB     C
 18:   A00F 3802                     JR      C,SKIP
 19:   A011 2C                       INC     L
 20:   A012 67                       LD      H,A
 21:   A013 10F7           SKIP:     DJNZ    LOOP
 22:   A015 C9                       RET
 23:
 24:   A016                          END
```

次の〝ADD　HL，HL〞で，Lレジスタの被除数の最位ビットをHレジスタに送っています。このループ中のHLレジスタの使いかたはちょっと複雑です。**図4-4**にHLレジスタの使いかたを図示してみました。初め，Hレジスタには0，Lレジスタには被除数が入っています。〝ADD　HL，HL〞を実行するたびに，Hレジスタ，Lレジスタのそれぞれの下位ビットから順に，余りと商が入ってきます。8回ループを繰り返すとHレジスタは余り，Lレジスタは商を持つことになります。

〝ADD　HL，HL〞でHレジスタに送られたものからCレジスタを引きます。このとき，引くことができれば〝INC　L〞でLレジスタの最下位ビットを1にします。〝ADD　HL，HL〞を実行するとLレジスタの最下位ビットは必ず0になるからです。

同様に，16ビット÷16ビットのサブルーチンは**リスト4-12**のようになります。ループ回数は16になるので，これにAレジスタを使っています。まず，〝SLA　E〃，〝RL　D〃，〝ADC　HL, HL〃で，DEレジスタの最上位ビットをHLレジスタに送ります(**図4-5**)。そして，HLレジスタから〝SBC　HL, BC〃で引き算します。HLレジスタは初めに0にしてあるので，〝ADC　HL, HL〃を実行後，常にキャリーフラグは0になります。そのため〝SBC　HL, BC〃の前でキャリーフラグをリセットする必要はありません。

引き算の後，キャリーフラグが1にならなければEレジスタの最下位ビットを1にし，キャリーフラグが1な

らば"ADD　HL, BC"でHLレジスタを元に戻します。
以上のループを16回繰り返します。

図4-5

リスト4-12

```
X1 Self Assembler  Rev 1.0    PAGE    1

 1:                       ;-----------------------------------------------
 2:                       ;       LIST    4-12
 3:                       ;
 4:    A000                       ORG     0A000H
 5:    A000 111027        MAIN:   LD      DE,10000
 6:    A003 01E803                LD      BC,1000
 7:    A006 CD0AA0                CALL    DIV16
 8:    A009 C9                    RET
 9:                       ;
10:                       ;       DE = DE / BC
11:                       ;       HL = DE MOD BC
12:                       ;
13:    A00A 210000        DIV16:  LD      HL,0
14:    A00D 3E10                  LD      A,16            ; Set loop counter
15:    A00F CB23          LOOP:   SLA     E
16:    A011 CB12                  RL      D
17:    A013 ED6A                  ADC     HL,HL
18:    A015 ED42                  SBC     HL,BC
19:    A017 3803                  JR      C,SKIP
20:    A019 1C                    INC     E
21:    A01A 1801                  JR      SKIP1
22:    A01C 09            SKIP:   ADD     HL,BC
23:    A01D 3D            SKIP1:  DEC     A
24:    A01E 20EF                  JR      NZ,LOOP
25:    A020 C9                    RET
26:
27:    A021                       END
```

4 10進数表示

割り算ルーチンを応用して,"ＨＬレジスタの値を10進数で表示する" というプログラムを紹介しましょう。このルーチンは, ゲーム・プログラムの点数表示などに使えるでしょう。

HLレジスタに入っている数値は最大で65535ですから, まず10000で割って商を表示, その余りを1000で割って商を表示……というように5回繰り返せばよいことになり

ます。

プログラムは，**リスト4-13**のようになります。13行目で割り算した結果は必ず１桁なのでＥレジスタのみに答えが返ってきます。これをＡレジスタに入れ，30Ｈを加えるのは，数値から数字のＡＳＣＩＩコード（たとえば３のASCIIコードは33Ｈ）に変換するためです。24行目で割り算ルーチンをコールしているのは，除数を10で割るためです。

リスト4-13

```
X1 Self Assembler  Rev 1.0    PAGE    1

  1:                    ;-----------------------------------------------
  2:                    ;       LIST    4-13
  3:                    ;
  4:   0013 =           ACCPRT  EQU     0013H
  5:   A000                     ORG     0A000H
  6:   A000 210852      MAIN:   LD      HL,21000
  7:   A003 CD07A0              CALL    DECOUT
  8:   A006 C9                  RET
  9:                    ;
 10:   A007 011027      DECOUT: LD      BC,10000
 11:   A00A EB          LOOP:   EX      DE,HL
 12:   A00B CD25A0              CALL    DIV16
 13:   A00E 7B                  LD      A,E
 14:   A00F C630                ADD     A,30H
 15:   A011 CD1300              CALL    ACCPRT
 16:   A014 79                  LD      A,C
 17:   A015 3D                  DEC     A
 18:   A016 C8                  RET     Z
 19:   A017 59                  LD      E,C
 20:   A018 50                  LD      D,B
 21:   A019 010A00              LD      BC,10
 22:   A01C E5                  PUSH    HL
 23:   A01D CD25A0              CALL    DIV16
 24:   A020 E1                  POP     HL
 25:   A021 4B                  LD      C,E
 26:   A022 42                  LD      B,D
 27:   A023 18E5                JR      LOOP
 28:                    ;
 29:                    ;       DIV16
 30:                    ;               DE = DE / BC
 31:                    ;               HL = DE MOD BC
 32:                    ;
 33:   A025 210000      DIV16:  LD      HL,0
 34:   A028 3E10                LD      A,16            ; Set loop counter
 35:   A02A CB23        DIVLP:  SLA     E
 36:   A02C CB12                RL      D
 37:   A02E ED6A                ADC     HL,HL
 38:   A030 ED42                SBC     HL,BC
 39:   A032 3803                JR      C,SKIP
 40:   A034 1C                  INC     E
 41:   A035 1801                JR      SKIP1
 42:   A037 09          SKIP:   ADD     HL,BC
 43:   A038 3D          SKIP1:  DEC     A
 44:   A039 20EF                JR      NZ,DIVLP
 45:   A03B C9                  RET
 46:
 47:   A03C                     END
```

第5章

IOCSとI/Oポート

5-1 IOCS
5-2 ワーク・エリア
5-3 I/Oポート

5-1 IOCS

IOCSは入出力用のサブルーチン群です。これらを使うことにより簡単に入出力をコントロールすることができます。次ページから，それぞれの使い方を解説しますが，このなかで，**レジスタ**とある項目は保存（値を変えないこと）するレジスタを示しています。値を変えてほしくないレジスタは前もってスタックにPUSHなどしておくとよいでしょう。

なお，解説中のASCIIコード，コントロール・コードなどは付録を参照してください。

キーボードからの入力

INPUTF(0003H)

機　能　1行入力

レジスタ　AF以外保存

入　力　**DE**…データ入力アドレス

出　力　**DE**…入力データ先頭アドレス

キャリーフラグ…0でリターン・キーおよびCTRL＋Jによる正常入力，1でBREAKおよびCTRL＋Dによるキャンセル。このときのAレジスタの値を見て，BREAKかCTRL＋Dかを判断する。

A…キャリーフラグ＝1のときのみ意味を持ち，BREAKなら03H，CTRL＋Dなら04Hがセットされる。

説　明　1行分スクリーン・エディットを行いDEレジスタで指定されたアドレスから1行分のデータを取り込む。このサブルーチンからリターンするには次の4種のキー入力による。

❶リターン・キーおよびCTRL＋Mによる正常入力

❷CTRL＋Jによる現在のカーソルより前のデータの正常入力

❸SHIFT＋BREAKおよびCTRL＋Cによる入力キャンセル

❹CTRL＋Dによる入力キャンセル

入力キャンセルの❸，❹の場合はDEレジスタで指されるアドレスの内容は変化しない。

BINPUT(015AH)

機　　能　1行入力

レジスタ　AF，DE以外保存

入　　力　**DE**…データ入力アドレス

出　　力　**DE**…入力データ先頭アドレス

キャリーフラグ…0でリターン・キーおよびCTRL＋Jによる正常入力，1でBREAKおよびCTRL＋Dによる入力キャンセル。

A…キャリーフラグ＝1のときのみ意味を持ち，BREAKなら03H，CTRL＋Dなら04Hがセットされる。

説　　明　INPUTFのサブルーチンと同様1行入力のサブルーチンだが，特別にBASICのINPUT文用に用意されたもので，入力開始行を行の第1行目として入力開始桁より前にあるメッセージは入力しない（入力開始桁までDEバッファを進めてリターンする）。

BRKCHK(004AH)

機　　能　SHIFT＋BREAKが押されたかの判断

レジスタ　AF以外保存

出　　力　ゼロ・フラグ…1のときSHIFT＋BREAKが押され，0のとき押されていない。

説　　明　SHIFT＋BREAKおよびCTRL＋Cが押されたときにゼロ・フラグが1になる。

INKEY$(001BH)

機　　能　1文字入力

レジスタ　AF以外保存

入　　力　A…INKEY$のモード値

出　　力　A…1文字入力したキーのASCIIコード

説　　明　サブルーチンを呼ぶ前にAレジスタにモードをセットする。その値により返ってきたときのAレジスタの意味や途中の動作が違う。

●モードの値が0FFHのとき

新しいキーが押されたときや，リピート・モードでリピートがONになるごとに押されているキーのASCIIコードを返す。その他のときは0を返す。

●モードの値が01Hのとき

カーソル待ちで1文字入力してそのASCIIコードを返す。リピート・モードではリピートしたキーのコードも返す。

●モードの値が00Hのとき

キーボードのサブCPUから送られるASCIIコード部8ビットのデータをそのまま返す。

●モードの値が02Hのとき

キーボードのサブCPUから送られるファンクション・コード部8ビット（**図5-1**）のデータをそのまま返す。

図5-1 ファンクション・コードのビット構成

(MSB)　　　　　　　　　　　　　　　　　　　　　(LSB)

	7	6	5	4	3	2	1	0
	ファンクション	キーデータが 有効　無効	リピート	GRAPH	CAPS	カナ	SHIFT	CTRL
“0”	●テンキー ●ファンクションキー ●TVキー ●カセット・キー	●データ・コード(8ビット)が有効である ヌル・コード“00”以外が送られてきたとき	●リピート・データである	●GRAPHキーが押されている	●CAPSキーが押されている (LOCKされている)	●カナキーが押されている (LOCKされている)	●SHIFT キーが押されている	●CTRLキーが押されている
“1”	●上記以外	●データ・コード(8ビット)が無効である ヌル・コード“00”が送られてきたとき	●1回目のデータである	●GRAPHキーが離されている	●CAPSキーが離されている	●カナキーが離されている	●SHIFT キーが離されている	●CTRLキーが離されている

画面,プリンタへの出力

ACCPRT(0013H)

機　　能　1文字出力
レジスタ　すべて保存
入　　力　**A**…出力するASCIIコード
説　　明　Aレジスタで示すASCIIコードの文字（20H～0FFH）を画面に表示する。コントロール・コード（00H～1FH）はそのコードの処理が実行される。

CTRLJB(0577H)

機　　能　コントロール・コード処理
レジスタ　AF，BC，DE，HL以外は保存
入　　力　**A**…コントロール・コード（00H～1FH）
説　　明　Aレジスタで指定されたコントロール・コードの処理を行う。ACCPRTで00H～1FHを出力したものと同じ。

ACCDIS(04C8H)

機　　能　1文字出力
レジスタ　すべて保存
入　　力　**A**…出力するASCIIコード
説　　明　ACCPRTと同様の1文字出力のサブルーチンだが，コントロール・コード（00H～1FH）も画面に表示して，コントロール・コードとしての処理はしない。

SPPRT(04BAH)

機　　能 スペース出力

レジスタ AF以外保存

説　　明 スペース（ASCIIコード20H）を画面に出力する。

TABPRT(04ABH)

機　　能 X座標が10の倍数までスペース出力

レジスタ AF以外保存

説　　明 X座標が10の倍数になるまでスペースを出力する。

CR1(04A7H)

機　　能 改行

レジスタ AF以外保存

説　　明 次の行の先頭へカーソルを移動する（改行する）。

CR2(04A3H)

機　　能 行の先頭でないなら改行

レジスタ AF以外保存

説　　明 現在のカーソル位置が行の先頭でないなら改行し，行の先頭なら改行しない。

PRINT(000BH)

機　　能　文字列のプリント

レジスタ　AF以外保存

入　　力　**DE**…文字列の先頭アドレス

説　　明　DEで示すアドレスから始まる文字列を画面に出力します。エンド・コード（文字列の最後に置くコード）は0（ヌル・コード）であり，それ以外はすべて処理しその他のコントロール・コード（01H～1FH）はそのコードの処理が実行される。**リスト5-1**を参照のこと。

リスト5-1

```
X1 Self Assembler  Rev 1.0    PAGE    1

 1:                       ;-----------------------------------------------
 2:                       ;         LIST     5-1
 3:                       ;         Print Out
 4:   000B =              PRINT     EQU       000BH
 5:
 6:   C000                          ORG      0C000H
 7:
 8:   C000 1107C0                   LD       DE,MES          ;DE..String Address
 9:   C003 CD0B00                   CALL     PRINT
10:   C006 C9                       RET
11:
12:   C007 0C             MES:      DEFB     0CH             ;Screen Clear Code
13:   C008 47524150                 DEFM     'GRAPHIC'
14:   C00F 00                       DEFB     0               ;End Code
15:
16:   C010                          END
```

ACCLPL(12DCH)

機　　能　プリンタへの1文字出力

レジスタ　F以外保存

入　　力　**A**…出力するASCIIコード

説　　明　Aレジスタで示すASCIIコードの文字をプリンタに出力する。

TABLPL(1315H)

機　　能 HTABのプリンタ出力
レジスタ AF以外保存
説　　明 水平タブ（HTAB）をプリンタに出力する。

CR1LPL(12D5H)

機　　能 LFのプリンタ出力
レジスタ AF以外保存
説　　明 改行（LF）をプリンタに出力する。

ACCPRP(1420H)

機　　能 画面またはプリンタへの１文字出力
レジスタ F，AF'以外保存
入　　力 **A**…出力するASCIIコード
説　　明 Aレジスタで示すASCIIコードの文字をFILOUT (1472H) が０のとき画面に，１のときプリンタに出力する。なお，FILOUTの内容はモニタのＰコマンドで０から１，１から０に変わる。

TABPPP(143CH)

機　　能 HTABを画面またはプリンタに出力
レジスタ AF以外保存
説　　明 水平タブ（HTAB）をFILOUT（1472H）が０のとき画面，１のときプリンタに出力する。

CR1PRP(1446H)

機　　能　改行コードを画面またはプリンタに出力

レジスタ　AF以外保存

説　　明　改行コードをFILOUT（1472H）が0のとき画面に，1のときプリンタに出力する。

PRINTP(142FH)

機　　能　文字列を画面またはプリンタに出力

レジスタ　AF，AF'以外保存

入　　力　**DE**…文字列の先頭アドレス

説　　明　DEで示すアドレスから始まる文字列をFILOUT(1472H)が0のとき画面，1のときプリンタに出力する。

表示のコントロール

WIDTH80(098CH)

機　　能　WIDTH80

レジスタ　AF, BC, DE, HL以外は保存

説　　明　80文字のモード指定をするとともに, IOCSのワーク(WIDTH 0 : 0007H) に80文字モードをセットする。この際, 画面はテキスト, グラフィック共にクリアされ, コンソールは解除され最大値となる。

ただし, グラフィックRAMの利用のためSCRMOD (0A8BH)が02Hならばグラフィック画面のクリアは行わない。

WIDTH40(0998H)

機　　能　WIDTH40

レジスタ　AF, BC, DE, HL以外は保存

説　　明　40文字のモード指定をするとともに, IOCSのワーク(WIDTH 0: 0007H)に40文字モードをセットする。この際, 画面はテキスト, グラフィック共にクリアされ, コンソールは解除され最大値となり, スクリーンも0ページ入出力のモードとなる。

ただし, SCRMOD (0A8BH) が02Hならばグラフィック画面のクリアは行わない。

CLST(0A6BH)

機　　能　テキスト・クリア

レジスタ　AF，BC，D，HL以外保存

説　　明　テキスト画面をクリアする。クリアするときの文字はCLSCHR(0027H)，アトリビュートはCOLORF (0026H) のワークにストアしている。

CLSG(0A8FH)

機　　能　グラフィック・クリア

レジスタ　AF，BC以外保存

説　　明　グラフィック画面を同時アクセス・モードでクリアする。

SCRNOT(09C0H)

機　　能　表示画面のページ指定

レジスタ　AF以外保存

入　　力　A…0でPAGE0を出力，1でPAGE1を出力。

説　　明　Aレジスタで示すページを出力ページとする。WIDTH40のモードでのみこのルーチンを呼ぶ。WIDTH80のモードでは何もしない（**リスト5-2**参照）。

リスト5-2

```
X1 Self Assembler  Rev 1.0    PAGE    1


  1:                    ;--------------------------------------------------
  2:                    ;        LIST    5-2
  3:                    ;        Page Set
  4:   0A8B =           SCRMOD  EQU      0A8BH
  5:   0998 =           WIDTH40 EQU      0998H
  6:   09C0 =           SCRNOT  EQU      09C0H          ;Set Display Screen
  7:   09F5 =           SCRNIN  EQU      09F5H          ;Set Write Screen
  8:
  9:   C000                     ORG     0C000H
 10:
 11:   C000 3E01                LD      A,1
 12:   C002 CDF509              CALL    SCRNIN          ;Write Page 1 Set
 13:   C005 3E01                LD      A,1
 14:   C007 CDC009              CALL    SCRNOT          ;Display Page 1 Set
 15:   C00A C9                  RET
 16:
 17:   C00B                     END
```

SCRNIN(09F5H)

機　　能　書き込み画面のページ指定

レジスタ　AF以外保存

入　　力　A…0でPAGE0にプリント，1でPAGE1にプリントする。

説　　明　Aレジスタで示すページにプリント処理されるようにIOCS内の出力用ワーク・エリアを設定する。このルーチンはWIDTH40のみ有効(**リスト5-2**参照)。

CTRLD?(0A3FH)

機　　能　コンソールと入出力ページの初期化

レジスタ　AF以外保存

説　　明　コンソール（テキスト画面の座標範囲）を最大値に戻し入出力のページを両方とも0ページに指定する。

STPRIO(0A5AH)

機　　能　パレット，プライオリティ・セット

レジスタ　AF，BC，D，HL以外保存

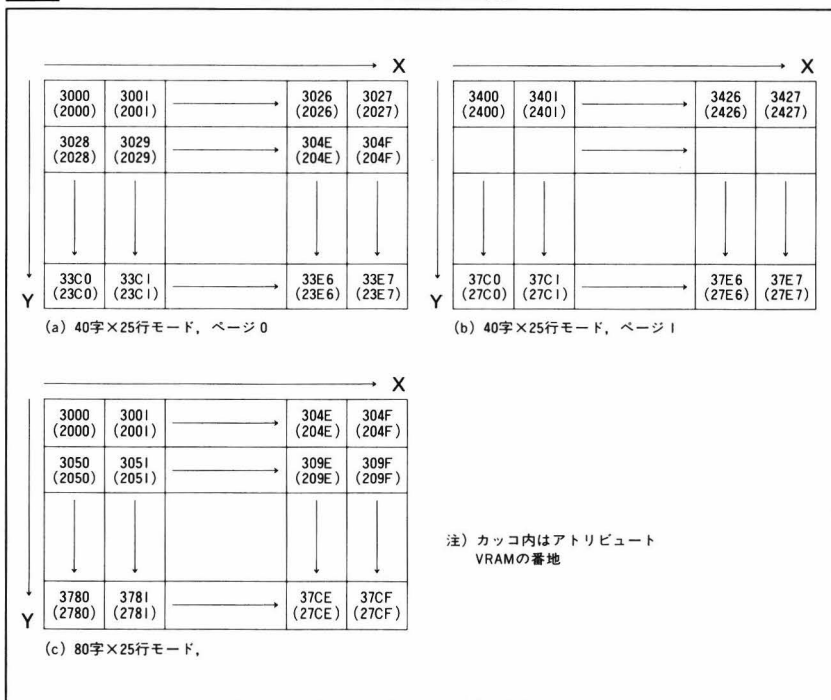
説　　明　RPRIOF(00F6H)青のパレット，GPRIOF(00F7H)赤のパレット，BPRIOF(00F8H)緑のパレット，TPRIOF(00F9H)プライオリティの各ワークの値をI/Oアドレスのパレット青（1000H），パレット赤(1100H)，パレット緑(1200H)，プライオリティ（1300H）にセットする（**リスト5-3**参照）。

リスト5-3

```
 X1 Self Assembler  Rev 1.0    PAGE    1


  1:                      ;---------------------------------------------
  2:                      ;        LIST     5-3
  3:                      ;        Back Color Set
  4:   00F6 =             RPRIOF   EQU       00F6H            ;Blue Pallet Work
  5:   0A5A =             STPRIO   EQU       0A5AH            ;Priority Set
  6:
  7:                      ;        Input    A..Code(0-7)
  8:
  9:   C900                        ORG      0C900H
 10:
 11:   C900 21F600                 LD       HL,RPRIOF
 12:   C903 0603                   LD       B,3
 13:   C905 CB1E          LOOP:    RR       (HL)
 14:   C907 0F                     RRCA
 15:   C908 CB16                   RL       (HL)
 16:   C90A 23                     INC      HL
 17:   C90B 10F8                   DJNZ     LOOP
 18:   C90D C35A0A                 JP       STPRIO           ;Priority Set
 19:
 20:   C910                        END
```

ADRCAL(054AH)

機　　能　テキストVRAMのアドレス計算
レジスタ　AF，BC，HL以降は保存
出　　力　**HL**…テキストVRAMのアドレス
説　　明　現在のカーソル位置のテキストVRAMのアドレスをHLレジスタに返す。アトリビュート・アドレスはHL-1000Hの値（**図5-2**）。

図5-2 テキストVRAMとアトリビュートVRAMの関係

ADRCA2(054DH)

機　　能　テキストVRAMのアドレス計算

レジスタ　AF，BC，HC以外保存

入　　力　**HL**テキスト座標(X，Y)の値(L，H)

出　　力　**HL**…テキストVRAMアドレス

説　　明　HLレジスタで指定したテキスト座標のテキストVRAMのアドレスをHLレジスタに返す。アトリビュート・アドレスはHL-1000Hの値。

カセット・コントロール

SAVE1(003BH)

機　　能　ファイル・コントロールブロック(FCB)をカセットテープにセーブ。

レジスタ　AF以外保存

入　　力　**HL**…FCBの先頭アドレス
BC…FCBの長さ(バイト数), X1では20Hを指定する

出　　力　**A** …0：セーブOK
1：BREAK
3：SET TAPE
4：WRITE PROTECT
5：その他, テープ・エンド およびカセット・キーが押された
キャリーフラグ…0：OK
1：ERROR

説　　明　ファイル・コントロールブロック(FCB)をカセットテープにセーブ。FCBの構造は**図5-3**を参照のこと(**リスト5-4**参照)。

リスト5-4

```
 X1 Self Assembler  Rev 1.0    PAGE    1


  1:                          ;-----------------------------------------------
  2:                          ;       LIST    5-4
  3:                          ;       Save Test
  4:                          ;Entry
  5:   0003 =                 INPUTF  EQU       0003H          ;Line Input
  6:   000B =                 PRINT   EQU       000BH
  7:   003B =                 SAVE1   EQU       003BH          ;Save FCB
  8:   003E =                 SAVE2   EQU       003EH          ;Save Body
  9:   0041 =                 LOAD1   EQU       0041H          ;Read FCB
 10:   0044 =                 LOAD2   EQU       0044H          ;Read Body
 11:   0047 =                 VERFY2  EQU       0047H          ;Verify Body
 12:   0DEC =                 CMTCOM  EQU       0DECH          ;CMT Control
 13:
 14:   9000                           ORG       9000H
 15:
 16:   9000 112C90                    LD      DE,WRTMES        ;Print 'Writing..'
```

```
17:   9003 CD0B00           CALL    PRINT
18:   9006 214490           LD      HL,FCB               ;Save FCB Address
19:   9009 012000           LD      BC,20H               ;FCB Length Set
20:   900C CD3B00           CALL    SAVE1                ;Save FCB
21:   900F 3810             JR      C,ERR
22:   9011 2100E0           LD      HL,0E000H            ;Save Start Address Set
23:   9014 010003           LD      BC,300H              ;Length Set
24:   9017 CD3E00           CALL    SAVE2                ;Save Body
25:   901A 3805             JR      C,ERR
26:   901C 113790           LD      DE,OKMES
27:   901F 1803             JR      PRT
28:   9021 113B90  ERR:     LD      DE,ERRMES
29:   9024 CD0B00  PRT:     CALL    PRINT
30:   9027 3E01    CMTSTP:  LD      A,1                  ;CMT STOP
31:   9029 C3EC0D           JP      CMTCOM
32:
33:   902C 57726974 WRTMES: DEFM    'Writing..
34:   9035 0D00             DEFB    0DH,0
35:   9037 4F4B    OKMES:   DEFM    'OK'
36:   9039 0D00             DEFB    0DH,0
37:   903B 4552524F ERRMES: DEFM    'ERROR !
38:   9042 0D00             DEFB    0DH,0
39:
40:   9044 01      FCB:     DEFB    1        ;00         Binary File
41:   9045 54               DEFM    'T'      ;01         File Name(13 Bytes)
42:   9046 45               DEFM    'E'      ;02
43:   9047 53               DEFM    'S'      ;03
44:   9048 54               DEFM    'T'      ;04
45:   9049 20               DEFB    20H      ;05
46:   904A 20               DEFB    20H      ;06
47:   904B 20               DEFB    20H      ;07
48:   904C 20               DEFB    20H      ;08
49:   904D 20               DEFB    20H      ;09
50:   904E 20               DEFB    20H      ;0A
51:   904F 20               DEFB    20H      ;0B
52:   9050 20               DEFB    20H      ;0C
53:   9051 20               DEFB    20H      ;0D
54:   9052 20               DEFB    20H      ;0E         EXT(3 Bytes)
55:   9053 20               DEFB    20H      ;0F
56:   9054 20               DEFB    20H      ;10
57:   9055 20               DEFB    20H      ;11         Pass Word
58:   9056 00               DEFB    00H      ;12         Length L
59:   9057 03               DEFB    03H      ;13                H
60:   9058 00               DEFB    00H      ;14         Satrt  L
61:   9059 E0               DEFB    0E0H     ;15                H
62:   905A 00               DEFB    00H      ;16         Exec   L
63:   905B E0               DEFB    0E0H     ;17                H
64:   905C 84               DEFB    84H      ;18         Year
65:   905D 75               DEFB    75H      ;19         Month,Day of the Week
66:   905E 20               DEFB    20H      ;1A         Day
67:   905F 02               DEFB    02H      ;1B         Time
68:   9060 41               DEFB    41H      ;1C         Minute(s)
69:   9061 20               DEFB    20H      ;1D         File Address(3 Bytes)
70:   9062 20               DEFB    20H      ;1E
71:   9063 20               DEFB    20H      ;1F
72:
73:   9064                  END
```

図5-3 FCBの構造

00
モード
→●ファイルの種類を表わす
1.フロッピィ・ディスクの場合
00はKILLされたファイル
FFは使用ディレクトリ・テーブルの終わり。
ビット0が1…Binファイル(マシン語で書かれたファイル)
ビット1が1…Basファイル(BASICテキストで書かれたファイル)
ビット2が1…Ascファイル(ASCIIセーブされたファイル)
ビット4が1…FILESで表示しない:0…表示する。
ビット5が1…リード・アフターライトON:0…OFF
ビット6が1…書き込み禁止ファイル:0…書き込みOK
ビット3とビット7は予備
(注)カセット,ROMの場合ビット0〜2までフロッピィ・ディスクと同じだが,ビット3〜7は未使用。
01
02
03
04
05
06
07
08
09
0A
0B
0C
0D
ファイル名
●ファイル名(13文字)
0E
0F
10
拡張子
●ユーザー指定拡張子エリア(3文字)
11
パスワード
→●パスワード無指定の場合20Hを入れる。
12
13
データ長
(L)
(H)
●プログラムの長さをバイト数で示す。
ただし,マシン語ファイルのみ有効となる。
14
15
データ先頭アドレス
(L)
(H)
●ロード時のメイン・メモリ先頭アドレスが入る。ただし,マシン語ファイルのみ有効。
16
17
実行アドレス
(L)
(H)
●ロードされたプログラムの実行開始番地をメイン・メモリ上のアドレスで指定。
18
19
1A
1B
1C
作成年月日
(年)
(月・曜日)
(日)
(時)
(分)
●ファイルが作成された年,月,曜日,日付,時刻(時,分)が入る。
1D
1E
1F
システム格納アドレス
●外部デバイス上のどこのアドレスから,ファイル本体が格納されているかを示す。カセットテープの場合常に00が格納される。

SAVE2(003EH)

機　　能　データをカセットテープにセーブ

レジスタ　AF以外保存

入　　力　**HL**…セーブするデータの先頭アドレス

BC…セーブするデータのバイト数

出　　力　**A**　…0：セーブOK

1：BREAK

3：SET TAPE

4：WRITE PROTECT

5：その他，テープ・エンドおよびカセット・キーが押された

キャリーフラグ…0：OK

1：ERROR

説　　明　HLで示されるメイン・メモリのアドレスからBCバイトをカセットにセーブする。BC＝0ならば65536バイト分セーブする（**リスト5-4**参照)。

LOAD1(0041H)

機　　能　ファイル・コントロールブロック（FCB）をカセットテープからロードする。

レジスタ　AF以外保存

入　　力　**HL**…FCBをロードしてくるメモリの先頭アドレス

BC…FCBの長さ（X1では20Hを指定）

出　　力　**A**　…0：ロードOK

1：BREAK

2：CHECK SUM ERROR

3：SET TAPE

5：その他，テープ・エンドおよびカセット・キーが押された。

キャリーフラグ…0：OK

1：ERROR

説　　明　HLで示すメイン・メモリのアドレスにカセットテープからファイル・コントロールブロック(FCB)をロードする（**リスト5-5**参照)。

リスト5-5

```
 X1 Self Assembler  Rev 1.0     PAGE     1

  1:                        ;-----------------------------------------------
  2:                        ;        LIST    5-5
  3:                        ;        Load Test
  4:                        ;Entry
  5:   0003 =               INPUTF  EQU      0003H          ;Line Input
  6:   000B =               PRINT   EQU      000BH
  7:   1321 =               FMPRHL  EQU      1321H          ;File Name Print
  8:   1394 =               SETDI?  EQU      1394H          ;File Name Set
  9:   134E =               FNMTCH  EQU      134EH          ;File Name Match
 10:   0041 =               LOAD1   EQU      0041H          ;Read FCB
 11:   0044 =               LOAD2   EQU      0044H          ;Read Body
 12:   0DEC =               CMTCOM  EQU      0DECH          ;CMT Control
 13:   F800 =               FCB     EQU     0F800H
 14:   04A3 =               CR2     EQU      04A3H
 15:
 16:   1480 =               DIRIMG  EQU      1480H          ;FCB
 17:   146A =               SKPMES  EQU      146AH          ;Skip Message
 18:   1462 =               FINMES  EQU      1462H          ;Found Message
 19:
 20:   0012 =               LENGTH  EQU      0012H
 21:
 22:   A000                         ORG     0A000H
 23:
```

```
24:   A000 1157A0            LD     DE,STMES          ;Print 'Please..'
25:   A003 CD0B00            CALL   PRINT
26:   A006 1100F9            LD     DE,0F900H
27:   A009 CD0300            CALL   INPUTF            ;Line Input
28:   A00C D8                RET    C                 ;Break End
29:   A00D CD9413            CALL   SETDI?            ;File Name Set
30:   A010 3E01              LD     A,1
31:   A012 328014            LD     (DIRIMG),A        ;Binary File Set
32:   A015 2100F8   MONLD:   LD     HL,FCB            ;FCB Address Set
33:   A018 012000            LD     BC,20H            ;FCB Length Set
34:   A01B CD4100            CALL   LOAD1             ;FCB Read
35:   A01E 3829              JR     C,ERR
36:   A020 CD4E13            CALL   FNMTCH            ;File Name Match
37:   A023 280D              JR     Z,MATCH
38:   A025 3E05              LD     A,5               ;APSS FF Set
39:   A027 CDEC0D            CALL   CMTCOM
40:   A02A 116A14            LD     DE,SKPMES         ;Print'Skip..'
41:   A02D CD2113            CALL   FMPRHL            ;File Name Print
42:   A030 18E3              JR     MONLD
43:                 ;
44:   A032 116214   MATCH:   LD     DE,FINMES         ;Print 'Found..'
45:   A035 CD2113            CALL   FMPRHL            ;File Name Print
46:   A038 2100B0            LD     HL,0B000H         ;Load Address Set
47:   A03B ED4B12F8          LD     BC,(FCB+LENGTH)
48:   A03F CD4400            CALL   LOAD2             ;Load Body
49:   A042 3805              JR     C,ERR
50:   A044 117CA0            LD     DE,OKMES          ;Print 'OK'
51:   A047 1803              JR     PRT
52:   A049 1180A0   ERR:     LD     DE,ERRMES         ;Print 'ERROR !'
53:   A04C CDA304   PRT:     CALL   CR2
54:   A04F CD0B00            CALL   PRINT
55:   A052 3E01     CMTSTP:  LD     A,1               ;CMT STOP
56:   A054 C3EC0D            JP     CMTCOM
57:
58:   A057 0D       STMES:   DEFB   0DH
59:   A058 506C6561          DEFM   'Please Input File Name '
60:   A06F 26205061          DEFM   '& Pass Word'
61:   A07A 0D00              DEFB   0DH,0
62:   A07C 4F4B     OKMES:   DEFM   'OK'
63:   A07E 0D00              DEFB   0DH,0
64:   A080 4552524F ERRMES:  DEFM   'ERROR !'
65:   A087 0D00              DEFB   0DH,0
66:
67:   A089                   END
```

LOAD2(0044H)

機　　能　カセットテープからデータをロード

レジスタ　AF以外保存

入　　力　**HL**…ロードするデータの先頭アドレス

　　　　　BC…ロードするデータの長さ

出　　力　**A** …0：ロードOK

　　　　　　　1：BREAK

　　　　　　　2：CHECK SUM ERROR

　　　　　　　3：SET TAPE

　　　　　　　5：その他，テープ・エンドおよびカセット・キーが押された

　　　　　キャリーフラグ…0：OK

　　　　　　　　　　　　1：ERROR

説　　明　HLで示すメイン・メモリのアドレスからBCバイトをカセットテープからロードする。BC＝0ならば65536バイト分ロードする(**リスト5-5**参照)。

VERFY2(0047H)

機　　能　カセットテープのデータとメイン・メモリのデータを比較する

レジスタ　AF以外保存

入　　力　**HL**…比較チェックする先頭アドレス
BC…比較するバイト数

出　　力　**A**　…0：比較OK
1：BREAK
2：CHECK SUM ERROR/VERIFY ERROR
3：SET TAPE
5：その他，テープ・エンドおよびカセット・キーが押された。
キャリーフラグ…0：OK
1：ERROR

説　　明　HLで示すメイン・メモリのアドレスからBCバイト分カセットテープのデータを比較チェックする。比較して内容が違っていればAレジスタに02Hの値を返す。

SETDI?(1394H)

機　　能　FCBにファイル・ネームをセット

レジスタ　AF，BC，DE，HL以外保存

入　　力　**DE**…ファイル・ネームが記憶してあるバッファの先頭アドレス

説　　明　DEで示すバッファよりファイル・ネームとパスワードDIRIMG（1480H）にあるFCBにセットする。このとき，パスワードはなくてもよい。また，ファイル・ネームをセットすると同時に日付けを読み出してFCBにセットする（**リスト5-5**参照）。

FMPRHL(1321H)

機　　能　FCBのファイル・ネームをプリント
レジスタ　AF，D以外保存
入　　力　**DE**…メッセージ出力先頭アドレス
　　　　　　HL…FCB先頭アドレス
説　　明　DEから始まるメッセージをプリントして，HLで示すFCBのファイル・ネームをプリントする（**リスト5-5**参照)。

FNMTCH(134EH)

機　　能　ファイル・ネームとパスワードを比較
レジスタ　AF以外保存
入　　力　**HL**…FCBの先頭アドレス
出　　力　**ゼロ・フラグ**…ゼロ・フラグが1のとき一致，0のとき不一致
説　　明　HLで示すFCBとDIRIMG（1480H）で示すFCBのファイル・ネームとパスワードを比較する。一致したらゼロ・フラグが1になり，不一致なら0になる（**リスト5-5**参照)。

CMTCOM(0DECH)

機　能　カセットに対しコマンドを実行する。

レジスタ　AF以外保存

入　力　**A**…カセット・コマンド値

説　明　Aレジスタにセットされているコマンドを実行する。以下にコマンド値を示す(**リスト5-6**参照)。

00H：EJECT
01H：STOP
02H：PLAY
03H：FF(FAST)
04H：REW
05H：APSS FF(APSS＋1)
06H：APSS REW(APSS－1)
0AH：REC

リスト5-6

```
X1 Self Assembler  Rev 1.0    PAGE    1


  1:                     ;---------------------------------------------
  2:                     ;        LIST    5-6
  3:                     ;        CMT Control
  4:   0013 =            ACCPRT   EQU       0013H
  5:   000B =            PRINT    EQU       000BH
  6:   04BA =            SPPRT    EQU       04BAH
  7:   001B =            INKEY    EQU       001BH
  8:   1327 =            FMPRHL   EQU       1327H          ;File Name Print
  9:   0041 =            LOAD1    EQU       0041H          ;FCB Read
 10:   0DEC =            CMTCOM   EQU       0DECH          ;CMT Control
 11:   F800 =            FCB      EQU      0F800H
 12:   04A3 =            CR2      EQU       04A3H
 13:
 14:   F000                       ORG      0F000H
 15:
 16:   F000 CD62F0                CALL     CLS             ;Screen Clear
 17:   F003 2103F0       L1:      LD       HL,L1
 18:   F006 E5                    PUSH     HL              ;Push Return Address
 19:   F007 CD57F0                CALL     CMTSTP          ;CMT STOP
 20:   F00A 116CF0                LD       DE,MES          ;Print Menu Message
 21:   F00D CD0B00                CALL     PRINT
 22:   F010 CD67F0                CALL     CHRGET          ;Get 1 Char
 23:   F013 CD1300                CALL     ACCPRT          ;Its Print
 24:   F016 FE31                  CP       '1'
 25:   F018 280D                  JR       Z,FILE          ;FILES
 26:   F01A FE32                  CP       '2'
 27:   F01C 2824                  JR       Z,APF           ;APSS FF
 28:   F01E FE33                  CP       '3'
 29:   F020 282A                  JR       Z,APR           ;APSS REW
 30:   F022 FE03                  CP       3
 31:   F024 C0                    RET      NZ              ;Goto L1
```

```
32:  F025 E1                 POP    HL
33:  F026 C9                 RET                    ;Return to Monitor
34:
35:  F027 11A4F0   FILE:     LD     DE,FILES        ;Print 'FILES'
36:  F02A CD0B00             CALL   PRINT
37:  F02D 2100F8   FILE1:    LD     HL,FCB          ;FCB Set
38:  F030 012000             LD     BC,20H          ;FCB Length Set
39:  F033 CD4100             CALL   LOAD1           ;FCB Read
40:  F036 381F               JR     C,CMTSTP
41:  F038 CD5CF0             CALL   FNMPR           ;File Name Print
42:  F03B 3E05               LD     a,5
43:  F03D CDEC0D             CALL   CMTCOM          ;APSS FF
44:  F040 18EB               JR     FILE1
45:
46:  F042 11ACF0   APF:      LD     DE,MAPF         ;Print 'APSS FF'
47:  F045 CD0B00             CALL   PRINT
48:  F048 3E05               LD     A,5             ;APSS FF
49:  F04A 1808               JR     CMT
50:
51:  F04C 11B6F0   APR:      LD     DE,MAPR         ;Print 'APSS REW'
52:  F04F CD0B00             CALL   PRINT
53:  F052 3E06               LD     A,6             ;APSS REW
54:  F054 CDEC0D   CMT:      CALL   CMTCOM
55:  F057 3E01     CMTSTP:   LD     A,1             ;CMT STOP
56:  F059 C3EC0D             JP     CMTCOM
57:
58:  F05C CDA304   FNMPR:    CALL   CR2
59:  F05F C32713             JP     FMPRHL
60:
61:                ;         Screen Clear
62:  F062 3E0C     CLS:      LD     A,0CH
63:  F064 C31300             JP     ACCPRT
64:
65:                ;         1 Char Get
66:  F067 3E01     CHRGET:   LD     A,1
67:  F069 C31B00             JP     INKEY
68:
69:  F06C          MES:      ;Menu Message
70:  F06C 0D                 DEFB   0DH
71:  F06D 312E2E46           DEFM   '1..FILES'
72:  F075 0D                 DEFB   0DH
73:  F076 322E2E41           DEFM   '2..APSS FF'
74:  F080 0D                 DEFB   0DH
75:  F081 332E2E41           DEFM   '3..APSS REW'
76:  F08C 0D                 DEFB   0DH
77:  F08D 50555348           DEFM   'PUSH (1-3) OR BREAK ? '
78:  F0A3 00                 DEFB   0
79:
80:  F0A4 0D       FILES:    DEFB   0DH
81:  F0A5 46494C45           DEFM   'FILES'
82:  F0AA 0D00               DEFB   0DH,0
83:
84:  F0AC 0D       MAPF:     DEFB   0DH
85:  F0AD 41505353           DEFM   'APSS FF'
86:  F0B4 0D00               DEFB   0DH,0
87:
88:  F0B6 0D       MAPR:     DEFB   0DH
89:  F0B7 41505353           DEFM   'APSS REW'
90:  F0BF 0D00               DEFB   0DH,0
91:
92:  F0C1                    END
```

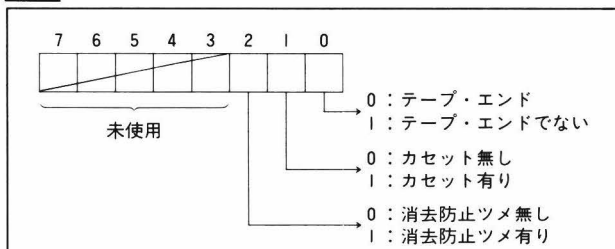
CMTSNS(0DF6H)

機　　能　カセットの状態を示す

レジスタ　AF以外保存

出　　力　A…カセットの状態

説　　明　カセットの状態をAレジスタの各ビットで返す。各ビットの意味は**図5-4**のようになる。

図5-4 カセットの状態

PSG

PSGINT(013CH)

機　　能　PSGの出力を止める

レジスタ　AF，BC，DE以外保存

説　　明　PSGの出力を止める。このときPSGのレジスタR７～R10に次のように値をセットする。

R_7 ←3FH

R_8 ←00H

R_9 ←00H

R_{10}←00H

BEEP(07F7H)

機　　能　ベル音を鳴らす

レジスタ　AF，BC，D以外は保存

説　　明　PSGを使ってベル音を鳴らす。

PCG

CGSET(002BH)

機　　能　PCGのセット

レジスタ　AF, E, HL以外保存

入　　力　D……ASCIIコード

E……セットI/Oアドレス上位(青:15H, 赤:16H, 緑:17H)

HL…セットするデータの先頭アドレス

出　　力　HL…データ先頭アドレス＋８の値

説　　明　PCGにパターンを定義する(**リスト5-7**参照)。

リスト5-7

```
X1 Self Assembler  Rev 1.0    PAGE    1


  1:                      ;---------------------------------------------
  2:                      ;         LIST    5-7
  3:                      ;         Define ASCII Code 40H
  4:   002B =             CGSET     EQU      002BH
  5:   0033 =             CGREAD    EQU      0033H
  6:
  7:   C000                         ORG     0C000H
  8:
  9:   C000 CD18C0                  CALL    CGCOPY              ;CG Copy(ROM->RAM
 10:   C003 214AC0                  LD      HL,CGDATA
 11:   C006 1640                    LD      D,40H
 12:   C008 1E15                    LD      E,15H               ;BLUE
 13:   C00A CD2B00                  CALL    CGSET
 14:   C00D 1E16                    LD      E,16H               ;RED
 15:   C00F CD2B00                  CALL    CGSET
 16:   C012 1E17                    LD      E,17H               ;GREEN
 17:   C014 CD2B00                  CALL    CGSET
 18:   C017 C9                      RET
 19:
 20:                      ;         CG Copy from ROM to RAM
 21:   C018 1600          CGCOPY:   LD      D,0
 22:   C01A 1E14          CGCPY1:   LD      E,14H
 23:   C01C 2142C0                  LD      HL,CGBUF
 24:   C01F CD32C0                  CALL    CGR                 ;ROM CG Read
 25:   C022 1C                      INC     E                   ;E=15H
 26:   C023 CD3AC0                  CALL    CGS                 ;BLUE
 27:   C026 1C                      INC     E                   ;E=16H
 28:   C027 CD3AC0                  CALL    CGS                 ;RED
 29:   C02A 1C                      INC     E                   ;E=17H
 30:   C02B CD3AC0                  CALL    CGS                 ;GREEN
 31:   C02E 15                      DEC     D
 32:   C02F 20E9                    JR      NZ,CGCPY1
```

```
33:    C031 C9                    RET
34:
35:    C032 D5         CGR:       PUSH    DE
36:    C033 E5                    PUSH    HL
37:    C034 CD3300                CALL    CGREAD          ;CG Read
38:    C037 E1                    POP     HL
39:    C038 D1                    POP     DE
40:    C039 C9                    RET
41:
42:    C03A D5         CGS:       PUSH    DE
43:    C03B E5                    PUSH    HL
44:    C03C CD2B00                CALL    CGSET           ;CG Set
45:    C03F E1                    POP     HL
46:    C040 D1                    POP     DE
47:    C041 C9                    RET
48:
49:    C042            CGBUF:     DEFS    8
50:
51:    C04A            CGDATA:
52:                    ;BLUE
53:    C04A 18                    DEFB    18H
54:    C04B 3C                    DEFB    3CH
55:    C04C 7E                    DEFB    7EH
56:    C04D FF                    DEFB    0FFH
57:    C04E FF                    DEFB    0FFH
58:    C04F 7E                    DEFB    7EH
59:    C050 3C                    DEFB    3CH
60:    C051 18                    DEFB    18H
61:                    ;RED
62:    C052 00                    DEFB    00H
63:    C053 00                    DEFB    00H
64:    C054 00                    DEFB    00H
65:    C055 3C                    DEFB    3CH
66:    C056 3C                    DEFB    3CH
67:    C057 00                    DEFB    00H
68:    C058 00                    DEFB    00H
69:    C059 00                    DEFB    00H
70:                    ;GREEN
71:    C05A E7                    DEFB    0E7H
72:    C05B C3                    DEFB    0C3H
73:    C05C 81                    DEFB    81H
74:    C05D 00                    DEFB    00H
75:    C05E 00                    DEFB    00H
76:    C05F 81                    DEFB    81H
77:    C060 C3                    DEFB    0C3H
78:    C061 E7                    DEFB    0E7H
79:
80:    C062                       END
```

CGREAD(0033H)

機　　能　キャラクタ・ジェネレータの内容を読む

レジスタ　AF, E, HL以外保存

入　　力　**D**……ASCIIコード

E……リードI/Oアドレス上位(14H～17H)

HL…データ用バッファの先頭アドレス

出　　力　**HL**…バッファ先頭アドレス＋８の値

説　　明　キャラクタ・ジェネレータ (CG) のパターンをメモリ内に読み込む。Eレジスタに設定する値は次のとおり (**リスト5-7**参照)。

E＝14H…ROM CG

E＝15H…RAM CG(PCG) BLUE

E＝16H…RAM CG(PCG) RED

E＝17H…RAM CG(PCG) GREEN

サブCPUとの通信

TRANS49(0B54H)

機　能　サブCPU8049に送信要求コードを送る

レジスタ　AF以外保存

入　力　**A**…8049に送る送信要求コード

説　明　Z80から8049に次のようなコントロールを行わせる。

1．キーコードの処理
2．モニタ画面モードの処理
3．TVの各種コントロール（チャンネル，音量他）
4．カセットのコントロール
5．時刻の設定，読み出し
6．タイマーの設定，読み出し

送信要求コードは**表5-1**のとおり（**リスト5-8**参照）。

リスト5-8

```
X1 Self Assembler  Rev 1.0    PAGE    1


  1:                  ;------------------------------------------------
  2:                  ;        LIST    5-8
  3:                  ;        Date & Time Set 1
  4:   0B54 =         TRANS49 EQU      0B54H
  5:
  6:   B900                    ORG     0B900H
  7:
  8:   B900 0608               LD      B,8
  9:   B902 110DB9             LD      DE,DATA
 10:   B905 1A        CD:      LD      A,(DE)
 11:   B906 13                 INC     DE
 12:   B907 CD540B             CALL    TRANS49
 13:   B90A 10F9               DJNZ    CD
 14:   B90C C9                 RET
 15:
 16:   B90D EC        DATA:    DEFB    0ECH             ;Date Set Code
 17:   B90E 84                 DEFB     84H             ;84 Year
 18:   B90F 73                 DEFB     73H             ;7 Month ,Wednesday
 19:   B910 19                 DEFB     19H             ;18 Day
 20:   B911 EE                 DEFB    0EEH             ;Time Set Code
 21:   B912 20                 DEFB     20H             ;20 Clock
 22:   B913 26                 DEFB     26H             ;26 Minutes
 23:   B914 30                 DEFB     30H             ;30 Seconds
 24:
 25:   B915                    END
```

表5-1 送信要求コード表

送信要求コード	内容	後続バイト数	方向 80C49 Z-80A	説明
E 4	キーベクタ値をセット	1	←	Z80のキーのベクタ・アドレスの下位バイトを返す。ただし，0の場合にはキー割り込み禁止モードとなる。
E 6	キーバッファ読み出し	2	←	8049のキーバッファの内容をZ80へ送る。キーバッファには最新のデータが格納されている。
E 7	TV送信コードセット	1	←	Z80より送信されてきたコードを8049のP27端子よりTVへ送る。
E 8	TV送信コード読み出し	1	→	TVに最後に送られたコードをZ80へ返す。
E 9	カセット指示	1	←	カセットメカの動作をする。
E A	カセット状態読み出し	1	→	カセットメカの動作状態を読み出しZ80へ返す。
E B	カセットセンサー読み出し	1	→	カセットセンサーを読み，その状態をZ80へ返す。
E C	日付けセット	3	←	時計用ICに"月"，"日"，"曜日"のデータを書き込む。
E D	日付け読み出し	3	→	時計用ICから"月"，"日"，"曜日"のデータを読み出しZ80へ返す。
E E	時刻セット	3	←	時計用ICに"時"，"分"，"秒"のデータを書き込む。
E F	時刻読み出し	3	→	時計用ICから"時"，"分"，"秒"のデータを読み出す。
D 0 ～ D 7	タイマー0(1, 2, 3, 4, 5, 6, 7)をセット(計8個)	6	←	タイマー0(1, 2, 3, 4, 5, 6, 7)の領域にデータを設定する。
D 8 ～ D F	タイマー0(1, 2, 3, 4, 5, 6, 7)を読み出す	6	→	タイマー0(1, 2, 3, 4, 5, 6, 7)の領域からデータを読み出す。

RECV49(0B49H)

機　　能　サブCPU8049からデータを受信

レジスタ　AF以外保存

出　　力　**A**…8049から受信したデータ

説　　明　8049からデータを受信してAレジスタにセットする。このサブルーチンはTRANS 49と共にZ 80と8049との通信に用いる（**リスト5-9, 5-10**参照）。

リスト5-9

```
 X1 Self Assembler  Rev 1.0     PAGE     1


  1:                     ;---------------------------------------------
  2:                     ;        LIST     5-9
  3:                     ;        Date & Time Set 2
  4:    0003 =           INPUTF   EQU       0003H           ;Line Input
  5:    000B =           PRINT    EQU       000BH
  6:    115E =           ACSET    EQU       115EH
  7:    0B54 =           TRANS49  EQU       0B54H
  8:    0B49 =           RECV49   EQU       0B49H
  9:    004A =           BRKCHK   EQU       004AH           ;Break Check
 10:
 11:    B000                      ORG      0B000H
 12:
 13:    B000 1123B0               LD       DE,MES          ;Print 'Date..'
 14:    B003 CD0B00               CALL     PRINT
 15:    B006 1135B0               LD       DE,BUF
 16:    B009 CD0300               CALL     INPUTF          ;Line Input
 17:    B00C D8                   RET      C               ;Break End
 18:    B00D 3EEC                 LD       A,0ECH          ;Date Set Code
 19:    B00F CD14B0               CALL     TSET
 20:    B012 3EEE                 LD       A,0EEH          ;Time Set Code
 21:    B014 CD540B      TSET:    CALL     TRANS49
 22:    B017 0603                 LD       B,3
 23:    B019 CD5E11      WSET:    CALL     ACSET           ;Acc Set
 24:    B01C D8                   RET      C               ;Error Return
 25:    B01D CD540B               CALL     TRANS49
 26:    B020 10F7                 DJNZ     WSET
 27:    B022 C9                   RET
 28:
 29:    B023 0D          MES:     DEFB     0DH
 30:    B024 44617465             DEFM     'Date & Time Set'
 31:    B033 0D00                 DEFB     0DH,0
 32:    B035             BUF:     DEFS     80
 33:
 34:    B085                      END
```

リスト5-10

```
X1 Self Assembler  Rev 1.0    PAGE    1


  1:                     ;---------------------------------------------
  2:                     ;        LIST    5-10
  3:                     ;        Date & Time Read
  4:                     ;Entry
  5:    04BA =           SPPRT    EQU      04BAH
  6:    1207 =           ACHXPR   EQU      1207H
  7:    0B54 =           TRANS49  EQU      0B54H
  8:    0B49 =           RECV49   EQU      0B49H
  9:    004A =           BRKCHK   EQU      004AH           ;Break Check
 10:
 11:                     ;Work
 12:    000E =           CX       EQU      000EH           ;Cursor X
 13:    000F =           CY       EQU      000FH           ;Cursor Y
 14:
 15:    9000                      ORG      9000H
 16:
 17:    9000 3EED        LOOP:    LD       A,0EDH          ;Date Read Code
 18:    9002 CD540B               CALL     TRANS49
 19:    9005 0603                 LD       B,3
 20:    9007 113C90               LD       DE,DTR
 21:    900A CD490B      DRD:     CALL     RECV49          ;Date read
 22:    900D 12                   LD       (DE),A
 23:    900E 13                   INC      DE
 24:    900F 10F9                 DJNZ     DRD
 25:    9011 3EEF                 LD       A,0EFH          ;Time Read Code
 26:    9013 CD540B               CALL     TRANS49
 27:    9016 0603                 LD       B,3
 28:    9018 113F90               LD       DE,DTR+3
 29:    901B CD490B      TRD:     CALL     RECV49          ;Time Read
 30:    901E 12                   LD       (DE),A
 31:    901F 13                   INC      DE
 32:    9020 10F9                 DJNZ     TRD
 33:    9022 0606                 LD       B,6             ;Print Date & Time
 34:    9024 213C90               LD       HL,DTR
 35:    9027 7E          PRT:     LD       A,(HL)
 36:    9028 23                   INC      HL
 37:    9029 CD0712               CALL     ACHXPR          ;Print Acc
 38:    902C CDBA04               CALL     SPPRT
 39:    902F 10F6                 DJNZ     PRT
 40:    9031 CD4A00               CALL     BRKCHK          ;Break Check
 41:    9034 C8                   RET      Z               ;If so Return
 42:    9035 AF                   XOR      A
 43:    9036 320E00               LD       (CX),A          ;Cursor X=0 Set
 44:    9039 18C5                 JR       LOOP
 45:    903B C9                   RET
 46:
 47:    903C             DTR:     DEFS     6
 48:
 49:    9042                      END
```

モニタ内サブルーチン

MONOP(0FE2H)

機　能　モニタへ制御を移す

説　明　モニタの各コマンドの使用を可能にする。BASICのMONコマンドの処理先。

HLHXPR(1202H)

機　能　HLレジスタの値を16進数で出力

レジスタ　AF，AF′以外保存

入　力　**HL**…出力する値

説　明　HLにセットされている値を16進数でFILOUT(1472H)が0のとき画面に，1のときプリンタに出力。たとえば，HLに1234Hがセットされていたら〝1234〟と表示する。

ACHXPR(1207H)

機　能　Aレジスタの値を16進数で出力

レジスタ　AF，AF′以外保存

入　力　**A**…出力する値

説　明　Aレジスタにセットする値を16進数でFILOUT(1472H)が0のとき画面，1のときプリンタに出力。

TUPPER(1451H)

機　　能　大文字に変換

レジスタ　AF以外保存

入　　力　**A**…変換したい文字コード

出　　力　**A**…変換された文字コード

説　　明　Aレジスタにセットされた文字を大文字に変換する。セットされた文字が英小文字（61H～7AH）のとき大文字に変換し，そうでないときはセットされた文字をそのまま返す。

HLSET(111FH)

機　　能　バッファ上の文字を16進数に変換してHLレジスタにセット

レジスタ　AF，DE，HL以外保存

入　　力　**DE**…バッファ・アドレス

出　　力　**HL**…キャリーフラグ＝0のときに有効な16進数がセットされて，キャリーフラグ＝1のときはHLは無変化

DE…キャリーフラグ＝0のとき，HLにデータをセットした分だけ進み，キャリーフラグ＝1のときDEは無変化

説　　明　このサブルーチンはACSETと同様に1行入力などのサブルーチン（たとえばINPUTFなど）と共に使えば，キーボードから入力した値をHLにセットできる（**リスト5-11**参照）。

リスト5-11

```
X1 Self Assembler  Rev 1.0     PAGE     1


 1:                       ;------------------------------------------------
 2:                       ;          LIST     5-11
 3:                       ;          Calculator
 4:                       ;Entry
 5:   0003 =              INPUTF    EQU       0003H          ;Line Input
 6:   0013 =              ACCPRT    EQU       0013H
 7:   000B =              PRINT     EQU       000BH
 8:   04A7 =              CR1       EQU       04A7H
 9:   04A3 =              CR2       EQU       04A3H
10:   1202 =              HLHXPR    EQU       1202H
11:   1207 =              ACHXPR    EQU       1207H
12:   111F =              HLSET     EQU       111FH
13:   115E =              ACSET     EQU       115EH
14:
15:                       ;Work
16:   0006 =              LINLIM    EQU       0006H
17:
18:   A000                          ORG      0A000H
19:
20:   A000 3E50                     LD       A,80
21:   A002 320600                   LD       (LINLIM),A
22:   A005 CDA704                   CALL     CR1
23:   A008 114FA0                   LD       DE,STMES           ;Print 'Calculator'
24:   A00B CD0B00                   CALL     PRINT
25:   A00E 1167A0                   LD       DE,BUFFER          ;Buffer Address Set
26:   A011 CD0300                   CALL     INPUTF             ;Line Input
27:   A014 D8                       RET      C                  ;Break End
28:   A015 CD1F11                   CALL     HLSET              ;HL Set
29:   A018 D8                       RET      C
30:   A019 2243A0                   LD       (AA),HL            ;+,-
31:   A01C 1A                       LD       A,(DE)
32:   A01D 13                       INC      DE
33:   A01E 3245A0                   LD       (OP),A
34:   A021 CD1F11                   CALL     HLSET
35:   A024 D8                       RET      C
36:   A025 EB                       EX       DE,HL
37:   A026 2A43A0                   LD       HL,(AA)
38:   A029 3A45A0                   LD       A,(OP)
39:   A02C FE2B                     CP       '+'
40:   A02E 2808                     JR       Z,PLUS
41:   A030 FE2D                     CP       '-'
42:   A032 C0                       RET      NZ
43:   A033 B7             MINUS:    OR       A
44:   A034 ED52                     SBC      HL,DE
45:   A036 1801                     JR       ANS
46:   A038 19             PLUS:     ADD      HL,DE
47:   A039                ANS:
48:   A039 1146A0                   LD       DE,MANS            ;Print 'Answer=..'
49:   A03C CD0B00                   CALL     PRINT
50:   A03F CD0212                   CALL     HLHXPR             ;HL..Print
51:   A042 C9                       RET
52:
53:   A043                AA:       DEFS     2
54:   A045                OP:       DEFS     1
55:
56:   A046 0D             MANS:     DEFB     0DH
57:   A047 416E7377                 DEFM     'Answer='
58:   A04E 00                       DEFB     0
59:
60:   A04F 43616C63 STMES:          DEFM     'Calculator Start (+,-)'
61:   A065 0D00                     DEFB     0DH,0
62:
63:   A067                BUFFER:   DEFS     81
64:
65:   A0B8                          END
```

ACSET(115EH)

機　　能　バッファ上の文字を16進数に変換してAレジスタにセット

レジスタ　AF，DE以外保存

入　　力　**DE**…バッファ・アドレス

出　　力　**A**　…キャリーフラグ＝0のときに有効な16進数がセットされて，キャリーフラグ＝1のときにはバッファ上のデータを返す。

DE…キャリーフラグ＝0のとき，Aレジスタにデータをセットした分だけ進み，キャリーフラグ＝1のとき，DEは無変化

説　　明　このサブルーチンは1行入力などのサブルーチン（たとえばINPUTFなど）と共に使えば，キーボードから入力した値をAレジスタにセットできる。使い方はHLSETと同じ。

5-2 ワーク・エリア

ここでは，IOCSやシステムで使っているワーク・エリアを解説します。注意しなければならないことは，書き換えて使えるものとできないものがあるということです。もちろんワーク・エリアもRAM上にあるわけですからすべて書き換えることも可能ですが，その場合の動作は保障できません。次からの解説では読み込み/書き換えができるものをR/W，読み込みしかできないものをＲで表わします。

LINLIM (0006H) R/W

意味 1行入力のバッファの長さのリミット

値 1～255

説明 INPUTF，BINPUTなどのスクリーン・エディットのサブルーチンで1行入力のとき，入力できる最大文字数を示す。この値以上の文字はバッファ内に読み込まれない。

WIDTHO (0007H) R

意味 WIDTHの値

値 40または80

説明 現在のスクリーンがWIDTH40か WIDTH80かを記憶しているバッファであり，40(28H)，80(50H)の値を持つ。このワークは読むだけで書き込んではいけない。

CURX (000EH) R/W

意味 カーソルのX座標

値 0～39 (WIDTH40)

0～79 (WIDTH80)

説明 現在のカーソルの位置のX座標を示す。書き変えるとカーソル位置を変更できる。

CURY (000FH) R/W

意味 カーソルのY座標

値 0～24

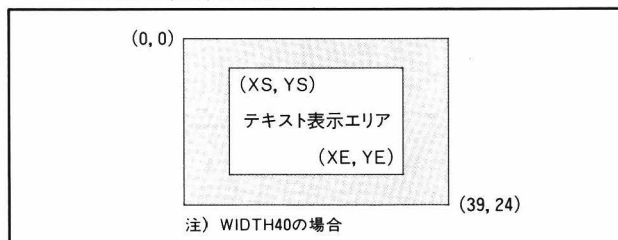
説明 現在のカーソル位置のY座標を示す。書き変えるとカーソル位置を変更できる。

CURYST (0016H) R/W

意味 テキスト表示エリアのY座標の先頭

値 0～24

説明 テキスト表示エリア (CONSOLE命令) のY座標のスタート座標を示す。**図5-5**参照。

図5-5 CURYST(YS), CURYED(YE), CURXST(XS), CURXED(XE)の関係

CURYED (0017H) R/W

意味 テキスト表示エリアのY座標の終わり

値 (CURYST)～24

説明 テキスト表示エリア (CONSOLE命令) のY座標のエンド座標を示す。

CURXST (001EH) R/W

意味 テキスト表示エリアのX座標の先頭

値 0～39 (または79)

説明 テキスト表示エリア (CONSOLE命令) のX座標のスタート座標を示す。

CURXED (001FH) R/W

意味 テキスト表示エリアのX座標の終わり

値 (CURXST)～39 (79)

説明 テキスト表示エリア (CONSOLE命令) のX座標のエンド座標を示す。

COLORF (0026H) R/W

意味 カラーアトリビュートの値

値 0～255

説明 ACCPRTなどでキャラクタを画面に表示するときのアトリビュートの値を示す。書き換え可能。
アトリビュートの値の各ビットの意味は**表5-2**のとおり。

表5-2 COLORF (0026H) のビット構成

ビット	説明			
0	青	COLOR	(0～7)	色指定
1	赤			
2	緑			
3		CREV	(0/1)	反転文字
4		CFLASH	(0/1)	点滅文字
5		CGEN	(0/1)	CG ROM/RAM
6		CSIZE	(0～3)	キャラスタ・サイズ
7				

(画面構成のアトリビュート参照)

CLSCHR (0027H) R/W

意味 NULキャラクタ・コード

値 通常20H

説明 スクリーン・エディットのときのNUL(ヌル)コードを示す。通常スペース (20H) が設定されている。たとえば2EHを書き込むとそれ以後，画面クリアもCTRL+EもZも〝.〟で埋まる。

KEYDAT (002EH, 002FH) R

意味 割り込みキー入力で入ってきたキーコード

説明 割り込みキー入力で最後に入ってきたキーの値がセットされる。

002EH…ASCIIコード

002FH…ファンクション・コード

BRKBUF (0036H) R

意味 割り込みキー入力によって入力されるBREAKおよびCTRL+Sを知らせるバッファ

説明 割り込みキー入力においてSHIFT+BREAKおよびBREAKが押された場合，このバッファに03Hおよび13Hが書き込まれてBREAKあるいは一時ストップの処理をする。このバッファをクリアするまでデータが残っている。

KEYFLG (0037H) R

意味 有効なキーが割り込んだときは0以外，無効なキーが割り込んだときは0の値を持つ。

説明 割り込みキー入力において有効なキーが割り込んだときは0以外，無効なキーが割り込んだときは0の値を持つ。キーが離された場合も0となる。

INIADR (00E9H, 00EAH)　R

意味　表示している画面のオフセット値

値　00E9H→00H, 04H
00EAH→00H

説明　表示している画面のオフセット値を示す(**表5-3**参照)。このワークの使い方はハード・コピープログラム（**リスト5-12**）を参照のこと。

表5-3 表示画面のオフセット値

WIDTH	ページ	00E9H	00EAH
40	0	00H	00H
	I	04H	00H
80	0	00H	00H

リスト5-12

```
 X1 Self Assembler  Rev 1.0     PAGE     1


   1:                       ;---------------------------------------------
   2:                       ;         LIST     5-12
   3:                       ;         HCOPY for Character
   4:                       ;Entry
   5:     12DC =            ACCLPL   EQU      12DCH          ;Printer Output
   6:     12D5 =            CR1LPL   EQU      12D5H
   7:     004A =            BRKCHK   EQU      004AH          ;Break Check
   8:
   9:                       ;Work
  10:     0007 =            WIDTH0   EQU      0007H          ;40 or 80
  11:     00E9 =            INIADR   EQU      00E9H          ;Offset
  12:
  13:     B800                       ORG     0B800H
  14:
  15:     B800 3AE900                LD      A,(INIADR)      ;Offset Read
  16:     B803 2E00                  LD      L,0
  17:     B805 67                    LD      H,A
  18:     B806 110030                LD      DE,3000H        ;Text Address
  19:     B809 19                    ADD     HL,DE
  20:     B80A E5                    PUSH    HL
  21:     B80B C1                    POP     BC              ;BC..Text Address
  22:     B80C CDD512                CALL    CR1LPL
  23:     B80F 1E19                  LD      E,25            ;Line Count Set
  24:     B811 3A0700       LL:      LD      A,(WIDTH0)
  25:     B814 57                    LD      D,A
  26:     B815 ED78         LOOP:    IN      A,(C)           ;Read ACSII Code
  27:     B817 03                    INC     BC
  28:     B818 CDDC12                CALL    ACCLPL          ;Printer Output
  29:     B81B CD4A00                CALL    BRKCHK          ;Break Check
  30:     B81E CAD512                JP      Z,CR1LPL
  31:     B821 15                    DEC     D
  32:     B822 20F1                  JR      NZ,LOOP
  33:     B824 CDD512                CALL    CR1LPL          ;CR
  34:     B827 1D                    DEC     E
  35:     B828 20E7                  JR      NZ,LL
  36:     B82A C9                    RET
  37:
  38:     B82B                       END
```

INIADW (00EBH, 00ECH) R

意味 アクセスしている画面のオフセット値

値 00EBH→00H

00ECH→00Hまたは04H

説明 アクセスしている（書き込む）画面のオフセット値を示す（**表5-4**，参照）。

表5-4 アクセス画面のオフセット値

WIDTH	ページ	00EBH	00ECH
40	0	00H	00H
	1	00H	04H
84	0	00H	00H

PRRIOF (00F6H) R

意味 青のパレット

値 初期値

説明 I/Oアドレス1000Hの値をもつワークで青のパレット状態を示す。

GPRIOF (00F7H) R

意味 赤のパレット

値 初期値

説明 I/Oアドレス1100Hの値をもつワークで赤のパレット状態を示す。

BPRIOF（00F8H） R

意味 緑のパレット

値 初期値

説明 I/Oアドレス1200Hの値をもつワークで緑のパレット状態を示す。

TPRIOF（00F9H） R

意味 テキストの優先順位

値 初期値 0

説明 I/Oアドレス1300Hの値をもつワークでテキストの優先順位を示す。
RPRIOF, GPRIOF, BPRIOF, TPRIOFは書き換えるとCTRLキー＋数字キーで背景色や文字グラフィックの色が望みの色に出ないことがある。

REPTF1（0366H）R/W

意味 リピートON/OFF

値 0, 1

説明 リピートのON/OFFのフラグで書き変え可能。0のときリピートOFFで1のときリピートON。

SCRMOD (0A8BH) R/W

意味 グラフィックのクリア指定

値 2, etc.

説明 WIDTH40, 80でグラフィックをクリアするかしないかを決定するフラグ。
2ならグラフィックをクリアしない。それ以外ならクリアする。

CLICKF (0E90H) R/W

意味 クリック音の制御フラグ

値 0, etc.

説明 割り込みキー入力時に入力確認音(クリック音)を出すか出さないかのフラグ。0ならクリック音を出し，それ以外なら出さない。

KBUFSW（0EA5H） R/W

意味　1行入力時，先行入力を捨てるかどうかのフラグ

値　0，etc.

説明　1行入力時，先行入力を捨てるかどうかのフラグで0の場合は捨てず，その以外の場合は捨てる。

POINT1（0EA6H） R/W

意味　INBUF内の書き込みポインタ

値　0～3FH

説明　先行入力およびファンクション・キー入力のためのINBUF内の書き込みポイントを示す。INBUFを参照のこと。

POINT2（0EA7H） R/W

意味　INBUF内読み込みポインタ

値　0～3FH

説明　先行入力およびファンクション・キー入力のためのINBUF内の読み込みポイントを示す。INBUFを参照のこと。

INBUF (0EA8H～0EE7H) R/W

意味 先行入力およびファンクション・キー入力のためのデータ・バッファ

説明 先行入力およびファンクション・キー入力のためのデータ・バッファ。POINT1およびPOINT2でそれぞれ書き込み，読み込みのポインタを表わしており，これはそのデータを持つワーク。

先行入力ルーチンにより次々とキーのデータがこのバッファに書き込まれ，書き込みポインタが進んでいく。このバッファはリング・バッファとなっているため，0EE7Hの次は0EA8Hに書くことになる。POINT1がPOINT2の一つ前にきている状態でバッファ・オーバーであり，それ以後のキーは受け取らない。

1文字入力ルーチンにより，このバッファより1バイトずつデータを読み込み，POINT2の読み込みポインタを進める。読み込みポインタと書き込みポインタが同じ値になったとき，バッファが空であることを意味する。

FUNBUF (0F42H～0FE1H) R/W

意味 ファンクション・キーが定義されているワーク

説明 ファンクション・キーの定義内容が書かれているワーク・エリア。ファンクション・キー1個につき16バイトのワークをもっており，10個で16×10=160バイトのワーク・エリア。

ワーク・エリアの構造は，最初の1バイトが定義バイト数(0～15)で残りの15バイトが定義文字。

FILOUT（1472H） R/W

意味 プリンタ・モード

値 0または1

説明 このワークはモニタのPコマンドで0または1に設定され，0のとき画面，1のときプリンタに出力装置を割り当てる。

DIRIMG（1480H）

意味 FCB（32バイト）

説明 モニタなどで使われるファイル・コントロールブロック（FCB）の先頭アドレス。

WRTMES（145AH） R

意味 ライティング・メッセージ

説明 カセットにデータをセーブするときのメッセージ。

FINMES（1462H） R

意味 ファインド・メッセージ

説明 カセットからファイルを見つけたときのメッセージ。

SKPMES（146AH） R

意味 スキップ・メッセージ

説明 カセットのファイルをスキップするときのメッセージ。

5-3　I/Oポート

Z80をCPUに使ったシステムでは，通常I/O空間を256バイト分しか持っていません。しかし，X1ではCレジスタを使った入出力命令（たとえばOUT (C), A）を実行したときにアドレス・バス上にBCレジスタの内容が出力され，これによって64KバイトのI/O空間を制御しています。グラフィックRAM（48Kバイト）もこのI/Oに接続されています（**図5-6**）。**表5-5**はシステムI/Oポートの詳細です。

図5-6 I/Oマップ

(a) シングルアクセス・モードI/Oマップ

I/Oアドレス	
0000–0FFF	ユーザーI/Oポート
1000–1FFF	システムI/Oポート
2000–27FF	属性 V･RAM
3000–37FF	テキスト V･RAM
4000–7FFF	グラフィック RAM1(B)
8000–BFFF	グラフィック RAM2(R)
C000–FFFF	グラフィック RAM3(G)

(b) 同時アクセス・モードI/Oマップ

I/Oアドレス	
0000–3FFF	グラフィック RAM(B･R･G)
4000–7FFF	グラフィック RAM(R･G)
8000–BFFF	グラフィック RAM(B･G)
C000–FFFF	グラフィック RAM(B･R)

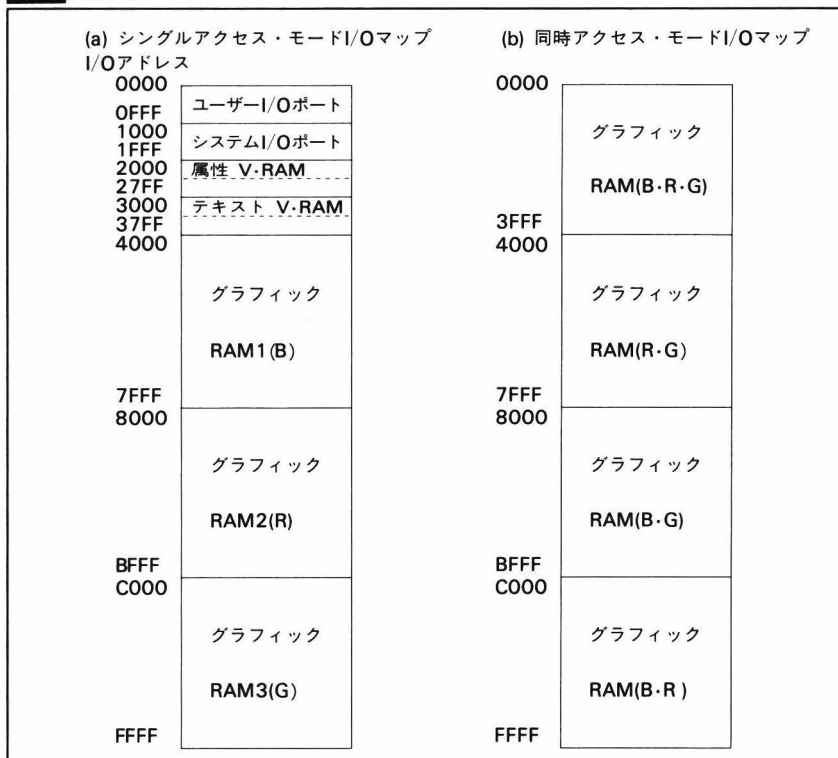



表5-5 システムI/Oポートの詳細

I/Oアドレス(16進)		内　　容	補　　足
Bレジスタ	Cレジスタ		
10	※※	パレットB	OUT命令のみ可
11	※※	パレットR	OUT命令のみ可
12	※※	パレットG	OUT命令のみ可
13	※※	プライオリティ	
14	※※	C/G ROM	
15	※※	C/G RAM B	
16	※※	C/G RAM R	
17	※※	C/G RAM G	
18	※0 ※1	CRTC	1800H：レジスタ 1801H：データ
19	※0 ※1 ※2	8255① (80C49, etc)	モード2 1900H：ポートA 1901H：ポートB 1902H：ポートC
1A	※0 ※1 ※2 ※3	8255② (プリンタ, etc)	モード0 1A00H：ポートA 1A01H：ポートB 1A02H：ポートC 1A03H：コントロール・レジスタ
1B	※※	PSGデータ	
1C	※※	PSGレジスタ	
1D	※※	IPLアクティブ	
1E	※※	IPLノンアクティブ	
1F	※※	未使用	

特に，グラフィックRAMは通常**図5-6(a)**のようにマッピング（割り当て）されていますが，グラフィックの処理を高速にするため，マッピングをハードウェアによって**図5-6(b)**のように一変させることが可能です。これらのマッピングをそれぞれ，シングルアクセス・モード，同時アクセス・モードと呼んでいます。

5-3-1 シングルアクセス・モード

シングルアクセス・モードはI/Oマップ上のデバイスに対して個別にアクセスするモードです。個別にアクセスするときは,

```
IN    A,(C)  ;ダミーリード
LD    BC,nn  ;nnはアクセスしたいI/Oアドレス
LD    A,n    ;nは書き込みたいデータ
OUT   (C),A
```

のように行います。

5-3-2 同時アクセス・モード

同時アクセス・モードはグラフィックのクリアや塗りつぶしを高速にするために設けられたモードです。たとえば,シングルアクセス・モードではグラフィックRAMの4000H, 8000H, C000Hのポートに同じデータを書き込みたいときに個別にアクセスしなければなりませんが,同時アクセス・モードでは0000Hにデータを書き込むだけで4000H, 8000H, C000Hに同時にデータの書き込みが行われます。同時アクセス・モードの設定は,

```
LD    BC, 1A03H  ┐
LD    A, 0BH     │ ビット5セット
OUT   (C), A     ┘
DEC   A          ┐
OUT   (C), A     ┘ ビット5リセット
LD    A, n       ;  nは書き込みたいデータ
LD    BC, nn     ;  nnは同時アクセスしたいI/O
OUT   (C), A                          アドレス
IN    A, (C)     ;  ダミーリード
```

というように行います。

シングル，同時アクセス・モードでダミーリードを行っているのは，これによって必ずシングルアクセス・モードに切り換わるシステム構成となっているからです。

5-3-3 テキスト画面

テキスト画面は，通常の文字（ASCIIコード）やプログラマブル・キャラクタ・ジェネレータ（PCG）に定義されたパターンを表示する画面で80字×25桁（1ページ）と40字×25桁(2ページ)の2つのモードを選択できます。

テキスト画面は，その属性（アトリビュート）VRAMを操作することで，8色の色指定，キャラクタ・ジェネレータ(CG)ROMかPCGかの切り替え，文字サイズ指定(縦倍，横倍，縦横倍)，反転，点滅を指定でき，これらすべてが1文字毎に8ビット・データとして対応しています。

テキスト画面はアトリビュートVRAMのアドレスと1対1に対応していて**図5-2**のような構成となっています。カッコ内の値がアトリビュートVRAMに割り当てられたアドレスで，その上の値がテキストVRAMのアドレスです。テキストVRAMのアドレスから1000H引いた値がア

リスト5-13

```
X1 Self Assembler  Rev 1.0    PAGE    1


  1:                     ;------------------------------------------------
  2:                     ;        LIST    5-13
  3:                     ;        Attribute Set
  4:
  5:    8000                      ORG      8000H
  6:
  7:    8000 012730               LD      BC,3027H          ;Text Address
  8:    8003 3E42                 LD      A,'B'             ;ASCII Code B(42H)
  9:    8005 ED79                 OUT     (C),A
 10:
 11:    8007 0620                 LD      B,20H             ;Attribute Address
 12:                                                        ;(BC=2027H)
 13:    8009 3E0F                 LD      A,0FH             ;Normal,CG ROM
 14:                                                        ;Reverse,White
 15:    800B ED79                 OUT     (C),A             ;Attribute Set
 16:    800D C9                   RET
 17:
 18:    800E                      END
```

トリビュート VRAM のアドレスです。アトリビュート VRAMのビットの内容は**表5-6**に示します。**リスト5-13**はサンプル・プログラムです。

表5-6 アトリビュートVRAMのビット内容

<table>
<tr><th>ビット番号</th><th>説　　明</th></tr>
<tr><td>0～2</td><td>キャラクタの色を指定します。カラー8色の表示が可能。
<table>
<tr><th colspan="3">ビット</th><th rowspan="2">指　定　色</th></tr>
<tr><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>黒</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>青</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>赤</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>赤紫(マゼンタ)</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>緑</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>水　色(シアン)</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>黄</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>白</td></tr>
</table>
</td></tr>
<tr><td>3</td><td>"1"→ビット0～2で指定したキャラクタ・カラーの補色表示(反転)を行う。</td></tr>
<tr><td>4</td><td>"1"→キャラクタを約0.5秒周期で点滅させます。</td></tr>
<tr><td>5</td><td>文字の表示モードを指定します。
<table>
<tr><th>ビット内容</th><th>設　定　モ　ー　ド</th></tr>
<tr><td>0</td><td>標準文字モード(CG ROM)</td></tr>
<tr><td>1</td><td>ユーザー文字モード(PCG)</td></tr>
</table>
</td></tr>
<tr><td>6～7</td><td>キャラクタの大きさを変えます。
<table>
<tr><th colspan="2">ビット</th><th rowspan="2">機　　能</th></tr>
<tr><th>7</th><th>6</th></tr>
<tr><td>0</td><td>0</td><td>ノーマル文字</td></tr>
<tr><td>0</td><td>1</td><td>垂直2倍文字</td></tr>
<tr><td>1</td><td>0</td><td>水平2倍文字</td></tr>
<tr><td>1</td><td>1</td><td>垂直・水平2倍文字</td></tr>
</table>
〔注〕ノーマル表示以外を指定する場合には，BASIC MANUAL p.77の注意を良く読んで使用のこと。
</td></tr>
</table>

5-3-4 グラフィック画面

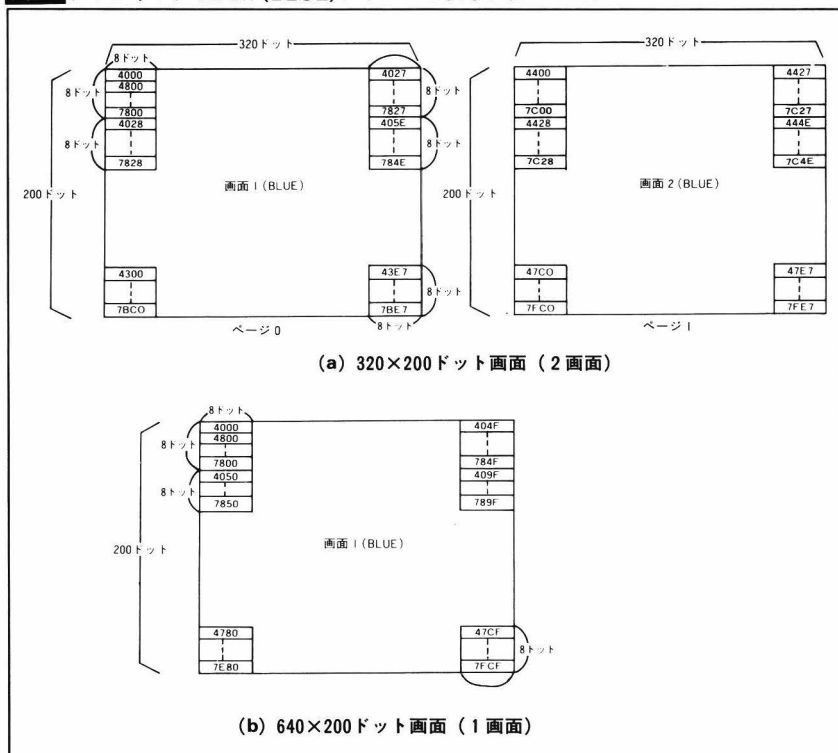
グラフィック画面は次の4つの何れかを選択できます。

❶640×200ドット……1面
❷320×200ドット……2面 } **ドット毎に8色指定**
❸640×200ドット……3面
❹320×200ドット……6面 } **面毎に8色指定**

グラフィックVRAM（BLUE）アドレスと表示位置との関係を**図5-7**に示します。REDやGREENにおいても同様なアドレス配置となります。BLUEのアドレスに4000H加えた値がREDのアドレスになり，さらにREDのアドレスに4000H加えた値がGREENのアドレスになります。

図5-7 グラフィックVRAM（BLUE）アドレスと表示位置との関係

(a) 320×200ドット画面（2画面）

(b) 640×200ドット画面（1画面）

グラフィック表示は320×200ドットと640×200ドットの2つのモードを選択することができます。これらのモード設定は，テキスト画面の40文字モードおよび80文字モード切り換えとまったく同じであり，テキスト画面が40文字モード時のグラフィック表示は320×200ドット(2ページ),80文字モード時のグラフィック表示は640×200ドット（1ページ）に対応しています。モードの設定は**リスト5-2**を参照してください。

表示画面を切り換えるには，パレットI/Oアドレス(**表5-5**）に**表5-7**のような設定データを書き込みます。

〈例〉 BLUE画面のみ表示

```
LD    BC, 1100H ┐
LD    A, 00H    │ RED画面を表示させない。
OUT   (C), A    ┘
INC   B         ┐
OUT   (C), A    ┘ GREEN画面を表示させない。
```

表5-7 表示画面とI/Oアドレスおよび設定データの関係

表示画面 ○：表示させる ×：表示させない			I/Oアドレスおよび設定データ		
GREEN	RED	BLUE	1200H	1100H	1000H
×	×	×	0 0	0 0	0 0
×	×	○	0 0	0 0	A A
×	○	×	0 0	C C	0 0
×	○	○	0 0	C C	A A
○	×	×	F 0	0 0	0 0
○	×	○	F 0	0 0	A A
○	○	×	F 0	C C	0 0
○	○	○	F 0	C C	A A

5-3-5 パレット機能

パレット機能とは，グラフィックRAMから出力される色をそのメモリのデータを書き換えせずに瞬時に別の色へ変えることができる機能です。

表5-8(a)を見てください。これはI/Oアドレスの1200H，1100H，1000HにそれぞれF0H，CCH，AAHの値がセットされてカラーコードで指定した色が画面で出ます(標準状態)。ここで，パレット・コード1をカラーコード5にセットするには**リスト5-14**のようにします（**表5-8(b)**参照)。

リスト5-15はパレットを設定するサブルーチンです。Dにパレット・コード，Eにカラーコードを入れてコールします。全レジスタの値は保存されます。

表5-8 カラーコードとパレット・コードのビット内容対応表

	パレット・コード		
カラー	G	R	B
コード	1200H	1100H	1000H
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
	F0H	CCH	AAH

(a) 標準状態

	パレット・コード		
カラー	G	R	B
コード	1200H	1100H	1000H
0	0	0	1
1	1	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
	F2H	CCH	AAH

(b) パレット・コード1をカラーコード5にセット

リスト5-14

```
X1 Self Assembler  Rev 1.0    PAGE    1


  1:                      ;-----------------------------------------------
  2:                      ;       LIST    5-14
  3:                      ;       Pallet 1->Color 5
  4:    1000 =            PLTB    EQU       1000H         ;Pallet Blue
  5:    1100 =            PLTR    EQU       1100H         ;Pallet Red
  6:    1200 =            PLTG    EQU       1200H         ;Pallet Green
  7:    1300 =            PRIO    EQU       1300H         ;Priority Port
  8:
  9:
 10:    C800                      ORG     0C800H
 11:
 12:    C800 010010               LD      BC,PLTB
 13:    C803 3EAA                 LD      A,0AAH          ;Pallet Blue Data
 14:    C805 ED79                 OUT     (C),A
 15:    C807 04                   INC     B               ;Pallet Red
 16:    C808 3ECC                 LD      A,0CCH          ;Pallet Red Data
 17:    C80A ED79                 OUT     (C),A
 18:    C80C 04                   INC     B               ;Pallet Green
 19:    C80D 3EF2                 LD      A,0F2H          ;Pallet Green Data
 20:    C80F ED79                 OUT     (C),A
 21:    C811 C9                   RET
 22:
 23:    C812                      END
```

リスト5-15

```
X1 Self Assembler  Rev 1.0    PAGE    1


  1:                      ;-----------------------------------------------
  2:                      ;       LIST    5-15
  3:    00F6 =            RPRIOF  EQU     00F6H           ;Pallet Work
  4:
  5:    1000 =            PALETB  EQU     1000H           ;Blue Pallet
  6:    1100 =            PALETR  EQU     1100H           ;Red Pallet
  7:    1200 =            PALETG  EQU     1200H           ;Green Pallet
  8:
  9:    A000                      ORG     0A000H
 10:
 11:                      ;       Pallet
 12:                      ;       Input   D..0FFH         Pallet Normal
 13:                      ;       or
 14:                      ;       Input   D..Pallet Code  Pallet Set
 15:                      ;               E..Color Code
 16:                      ;
 17:    A000 C5           PALLET: PUSH    BC
 18:    A001 D5                   PUSH    DE
 19:    A002 E5                   PUSH    HL
 20:    A003 F5                   PUSH    AF
 21:    A004 7A                   LD      A,D
 22:    A005 FEFF                 CP      0FFH
 23:    A007 283B                 JR      Z,PLTN
 24:    A009 7A                   LD      A,D
 25:    A00A CD51A0               CALL    BSET
 26:    A00D 57                   LD      D,A
 27:    A00E 7B                   LD      A,E
 28:    A00F CD51A0               CALL    BSET
 29:    A012 5F                   LD      E,A
 30:    A013 21F600               LD      HL,RPRIOF
 31:    A016 015EA0               LD      BC,PDATA
 32:    A019 E5                   PUSH    HL
 33:    A01A 3E03                 LD      A,3
 34:    A01C F5           PLTLP:  PUSH    AF
 35:    A01D 0A                   LD      A,(BC)
 36:    A01E A3                   AND     E
 37:    A01F 7A                   LD      A,D
 38:    A020 2004                 JR      NZ,PLT1
 39:    A022 2F                   CPL
```

```
40:    A023 A6                 AND     (HL)            ;Bit OFF
41:    A024 1801               JR      PLT2
42:    A026 B6         PLT1:   OR      (HL)            ;Bit ON
43:    A027 77         PLT2:   LD      (HL),A          ;Set Pallet Data
44:    A028 23                 INC     HL
45:    A029 03                 INC     BC
46:    A02A F1                 POP     AF
47:    A02B 3D                 DEC     A
48:    A02C 20EE               JR      NZ,PLTLP
49:    A02E E1                 POP     HL
50:                    ;       Pallet Data Set
51:    A02F 21F600     PLTST:  LD      HL,RPRIOF
52:    A032 010010             LD      BC,PALETB       ;Blue Pallet
53:    A035 1603               LD      D,3
54:    A037 7E         PLTLP2: LD      A,(HL)
55:    A038 ED79               OUT     (C),A
56:    A03A 23                 INC     HL
57:    A03B 04                 INC     B
58:    A03C 15                 DEC     D
59:    A03D 20F8               JR      NZ,PLTLP2
60:    A03F F1                 POP     AF
61:    A040 E1                 POP     HL
62:    A041 D1                 POP     DE
63:    A042 C1                 POP     BC
64:    A043 C9                 RET
65:
66:    A044            PLTN:   ;Init Pallet Data Copy
67:    A044 215EA0             LD      HL,PDATA
68:    A047 11F600             LD      DE,RPRIOF
69:    A04A 010300             LD      BC,3
70:    A04D EDB0               LDIR
71:    A04F 18DE               JR      PLTST
72:
73:    A051 B7         BSET:   OR      A               ;A Bit ON
74:    A052 2807               JR      Z,BST1
75:    A054 47                 LD      B,A
76:    A055 3E01               LD      A,1
77:    A057 87         BSTLP:  ADD     A,A
78:    A058 10FD               DJNZ    BSTLP
79:    A05A C9                 RET
80:
81:    A05B 3E01       BST1:   LD      A,1
82:    A05D C9                 RET
83:
84:    A05E AACCF0     PDATA:  DEFB    0AAH,0CCH,0F0H  ;Init Pallet Data
85:
86:    A061                    END
```

5-3-6 プライオリティ機能

X1の特徴として，テキスト画面をグラフィック画面の各色との間で優先順位を指定できる機能（プライオリティ機能）があります。

初期設定値として，プライオリティのI/Oアドレス(1300H)には00Hが設定されており，この状態は，テキスト画面がバック色やグラフィック画面に対して優先しています。

優先順位の組み合わせは，$2^8=256$とおり指定できます

(**表5-9**参照)。

優先順位の変更は，I/Oアドレス (1300H) を介して行います。たとえば，テキスト画面の表示文字をグラフィックのカラーの青，マゼンタ，シアン，黄の陰にしたい場合，**表5-9**より $(01101010)_2$ = 6 AHをセットすればよいことになります。

〈例〉優先順位の変更

```
LD    BC, 1300H  ;  プライオリティのI/Oポート
LD    A, 6AH     ┐
OUT   (C), A     ┘  データ・セット
```

リスト5-16は，8×8ドットのパターンとプライオリティをセットしたサンプル・プログラムです。

表5-9 プライオリティ (I/Oアドレス1300H) の組み合わせ

ビット	内　容
7	0：テキストは白より優先 1：白はテキストより優先
6	0：テキストは黄より優先 1：黄はテキストより優先
5	0：テキストはシアンより優先 1：シアンはテキストより優先
4	0：テキストは緑より優先 1：緑はテキストより優先
3	0：テキストはマゼンタより優先 1：マゼンタはテキストより優先
2	0：テキストは赤より優先 1：赤はテキストより優先
1	0：テキストは青より優先 1：青はテキストより優先
0	0：テキストはバック色より優先 1：バック色はテキストより優先

リスト5-16

```
X1 Self Assembler  Rev 1.0    PAGE    1


  1:                    ;---------------------------------------------
  2:                    ;        LIST     5-16
  3:                    ;        Graphic  8*8 dot
  4:    0A8F =          CLSG     EQU      0A8FH          ;Graphic Clear
  5:    1000 =          PLTB     EQU      1000H          ;Pallet Blue
  6:    1100 =          PLTR     EQU      1100H          ;Pallet Red
  7:    1200 =          PLTG     EQU      1200H          ;Pallet Green
  8:    1300 =          PRIO     EQU      1300H          ;Priority Port
  9:
 10:    4000 =          BLUE     EQU      4000H
 11:    8000 =          RED      EQU      8000H
 12:    C000 =          GREEN    EQU      0C000H
 13:
 14:    C000                     ORG      0C000H
 15:
 16:    C000 CD8F0A              CALL     CLSG           ;Graphic Clear
 17:    C003 010010              LD       BC,PLTB        ;Pallet Blue I/O
 18:    C006 3EAA                LD       A,0AAH         ;Pallet Blue Data
 19:    C008 ED79                OUT      (C),A
 20:    C00A 04                  INC      B              ;Pallet Red
 21:    C00B 3ECC                LD       A,0CCH         ;Pallet Red Data
 22:    C00D ED79                OUT      (C),A
 23:    C00F 04                  INC      B              ;Pallet Green
 24:    C010 3EF0                LD       A,0F0H         ;Pallet Green Data
 25:    C012 ED79                OUT      (C),A
 26:    C014 04                  INC      B              ;Priority
 27:    C015 AF                  XOR      A              ;Priority Data
 28:    C016 ED79                OUT      (C),A          ;Priority Set
 29:
 30:    C018 110008              LD       DE,800H
 31:    C01B DD213FC0            LD       IX,DATA
 32:
 33:    C01F 210040              LD       HL,BLUE
 34:    C022 CD2EC0              CALL     WRTG           ;Blue Data Set
 35:    C025 210080              LD       HL,RED
 36:    C028 CD2EC0              CALL     WRTG           ;Red Data Set
 37:    C02B 2100C0              LD       HL,GREEN
 38:    C02E 0608       WRTG:    LD       B,8            ;Green Data Set
 39:    C030 C5         LOOP:    PUSH     BC
 40:    C031 44                  LD       B,H
 41:    C032 4D                  LD       C,L
 42:    C033 DD7E00              LD       A,(IX)
 43:    C036 DD23                INC      IX
 44:    C038 ED79                OUT      (C),A
 45:    C03A 19                  ADD      HL,DE
 46:    C03B C1                  POP      BC
 47:    C03C 10F2                DJNZ     LOOP
 48:    C03E C9                  RET
 49:
 50:    C03F            DATA:
 51:    C03F 71         DATAB:   DEFB     71H
 52:    C040 71                  DEFB     71H
 53:    C041 71                  DEFB     71H
 54:    C042 71                  DEFB     71H
 55:    C043 71                  DEFB     71H
 56:    C044 01                  DEFB     01H
 57:    C045 01                  DEFB     01H
 58:    C046 01                  DEFB     01H
 59:
 60:    C047 01         DATAR:   DEFB     01H
 61:    C048 44                  DEFB     44H
 62:    C049 01                  DEFB     01H
 63:    C04A 44                  DEFB     44H
 64:    C04B 01                  DEFB     01H
 65:    C04C 44                  DEFB     44H
 66:    C04D 01                  DEFB     01H
 67:    C04E 44                  DEFB     44H
 68:
 69:    C04F 08         DATAG:   DEFB     08H
```

```
70:   C050 22          DEFB    22H
71:   C051 08          DEFB    08H
72:   C052 22          DEFB    22H
73:   C053 08          DEFB    08H
74:   C054 22          DEFB    22H
75:   C055 08          DEFB    08H
76:   C056 22          DEFB    22H
77:   C057             END
```

5-3-7 PSG(AY-3-8910)

PSG(Programmable Sound Generator) AY-3-8910は，3和音・8オクターブ出力可能なほかに，2つの8ビット出力ポートA・Bをもち，X1ではジョイスティックの入力用に使っています。

PSGは内部に16個の8ビット・レジスタ（**表5-10**）を持ち，これらにデータを書き込むことで，音の周波数(音程)，音量などをコントロールします。

●レジスタR_0〜R_5

PSGのレジスタのうち，R_0からR_5まではトーン・ジェネレータで発振される周波数をコントロールするのに使われています。PSGは3つのトーン・ジェネレータを持ち，それぞれをチャンネルA, B, Cと呼びます。

レジスタは各チャンネルに対しR_0とR_1, R_2とR_3, R_4とR_5をそれぞれペアにして使用し，そこに下位8ビット，上位4ビットの計12ビットのデータを書き込みます。チャンネルAを例にとると，R_0に書き込まれた8ビット・データと，R_1のビット0から3での間に書き込まれた4ビットのデータの合計12ビットのデータが発振周波数を決定します。それ以外のR_1のビット4から7までの4ビット分については，たとえ書き込まれても無視されます。

さて発振される周波数（音の高さ）fとレジスタに書きこんだ12ビットのデータDとの関係は次の式で表わすことができます。

$$f=\frac{fclock}{16\times D}$$

fclock：PSGへのクロック入力（2MHz）
f　　　：出力される周波数
D　　　：書き込まれる12ビットのデータの値

X1の場合，fclockの値は2MHzということになっているので，実際に出せる最高の音はデータの値が1のとき125KHzで，最低の音は4095のときの30.5Hzです。これは可聴帯域の音のすべてをカバーしています。

表5-10 PSGのレジスタ一覧

レジスタ	機能	7	6	5	4	3	2	1	0	
R_0	チャンネルAの周波数	下位8ビット・データ								
R_1						上位4ビット・データ				
R_2	チャンネルBの周波数	下位8ビット・データ								
R_3						上位4ビット・データ				
R_4	チャンネルCの周波数	下位8ビット・データ								
R_5						上位4ビット・データ				
R_6	ノイズの平均周波数				5ビット・データ					
R_7	ミキシング，ポート制御	入出力選択		ノイズ			トーン			対応するビットが1のときオフ，0のときオン
		ポートB	ポートA	C	B	A	C	B	A	
R_8	チャンネルAの音量				M	4ビット・データ				M＝0のとき下位4ビット・データによる音量調節 M＝1のとき エンベロープ作動
R_9	チャンネルBの音量				M	4ビット・データ				
R_{10}	チャンネルCの音量				M	4ビット・データ				
R_{11}	エンベロープ周期	下位8ビット・データ								
R_{12}		上位8ビット・データ								
R_{13}	エンベロープ波形					CONT	ATT	ALT	HOLD	
R_{14}	ポートA									R_7の入出力選択が1のとき出力，0のとき入力
R_{15}	ポートB									

●レジスタR_6

レジスタR_6はノイズの平均周波数をコントロールするものです。R_6に書き込まれたデータは，下位5ビットのみが有効で，上位3ビットについては無視されます。したがってR_6を使用する場合に，実際に有効となる値は1から31までということになります。ここでノイズの平均周波数をf，R_6に書き込まれるデータをDとすると，

$$f=\frac{fclock}{16\times D}$$

fclock：PSGへのクロック入力（2MHz）
f　　　：出力されるノイズの平均周波数
D　　　：R_6の下位5ビットの値

といった関係が成り立ちます。実際に出力されるノイズの平均周波数は，書き込むデータが1のときに125KHz 31のときに4.0KHzで，データが小さいときはホワイト・ノイズ（サーという音）のように聞こえ，逆に大きくなるとピンク・ノイズ（ゴーという風のような音）に近くなります。

●レジスタR_7

トーンとノイズのミキシングをする他，PSGの持つ2つのポートの入出力の方向を決定します。

このレジスタR_7のビット0から2までがトーンの出力を決定するスイッチで，ビット3から5までが同様にノイズの出力を決めるスイッチです。それぞれのビットに0を書き込むとONになり，1を書き込むとOFFになります。たとえばチャンネルBをトーンとノイズのミックス出力にしたい場合は，ビット4と1にそれぞれ0を書きこめばよいわけです。

さてまだ説明していないビット6とビット7の使い方ですが，これは入出力ポートの方向を決定するのに使用されています。X1では，入出力ポートがジョイスティッ

ク端子の入出力に使われており，ビット6とビット7の設定値は**表5-11**のようになっています。したがって，特別に用途がない限り，ビット6とビット7は0にしなければなりません。

表5-11 レジスタR_7のビット6，ビット7

ビット7	ビット6	入出力制御
0	0	ジョイスティック端子（JS）1・2共に入力
0	1	JS2：入力，JS1：出力
1	0	JS2：出力，JS1：入力
1	1	JS1・2共に出力

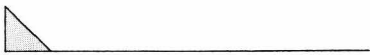
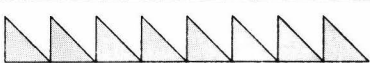
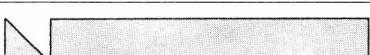
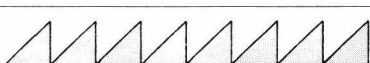
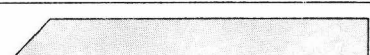
●レジスタR_8〜R_{10}

A, B, Cの各チャンネルの音量をコントロールします。音量は4ビットで表わし，15が最大で0が最少（無音）です。ただし，各レジスタのビット4を1にすると音量の調節はエンベロープ・ジェネレータに任されることになり，その場合は下位4ビットに設定した音量は無視されます。したがってエンベロープを使用しないときは，ビット4は単に0にしておく必要があります。またどの場合でも各レジスタ上位3ビットは無視されます。

●レジスタR_{11}，R_{12}

エンベロープの周期（**表5-12**のtに相当する時間）を決定します。R_{11}とR_{12}をペアにして16ビットとして使い，R_{11}に下位8ビット，R_{12}に上位8ビットのデータを書き込みます。周期tと16ビット・データDの関係は次の式で与えられます。

$$t=\frac{256\times D}{fclock}$$

fclock ：PSGへのクロック入力（2MHz）
t ：エンベロープ周期
D ：R_{11}とR_{12}に書き込まれた16ビット・データ

X1の場合，周期の最少が128マイクロ秒，最大が8.4秒となります。

●レジスタR_{13}

エンベロープのパターンを決定します。パターンの選択は，R_{13}の下位4ビットに書き込まれるデータで**表5-12**のように決まります。R_{13}にデータを書き込むと同時にエンベロープのトリガーがかかるので，再度トリガー※をかけたい場合は，そのたびにパターンを指定し直さなければなりません。

※トリガー
もともと引き金の意味で，ここではエンベロープをかけるきっかけとなる信号のこと。

表5-12 エンベロープ・パターン

レジスタR_{13}の下位4ビット	16進値	エンベロープ・パターン
0 0 ー ー	0～3	
0 I ー ー	4～7	
I 0 0 0	8	
I 0 0 I	9	
I 0 I 0	A	
I 0 I I	B	
I I 0 0	C	
I I 0 I	D	
I I I 0	E	
I I I I	F	

t

●レジスタR_{14}，R_{15}

R_{14}，R_{15}とも８ビットの入出力ポートです。ジョイスティックの入出力はこのポートを介して行います。

PSGへアクセスする方法は，まずPSGのレジスタ番号をセットし，さらにデータをセットします。**リスト5-17**は，BASICのSOUND文に相当するもので，AレジスタにPSGのレジスタ番号，Dレジスタにデータを入れてコールします。**リスト5-18**は，逆にPSGからデータを読み込むサブルーチンです。

リスト5-17

```
X1 Self Assembler  Rev 1.0    PAGE     1


  1:                       ;-----------------------------------------------
  2:                       ;       LIST     5-17
  3:                       ;       Sound Set(PSG)
  4:                       ;       Input    A..Register
  5:                       ;                D..Data
  6:    1C00 =             PSGCOM  EQU       1C00H          ;PSG Register I/O Port
  7:    1B00 =             PSGDAT  EQU       1B00H          ;PSG Data I/O Port
  8:
  9:    C000                       ORG      0C000H
 10:
 11:    C000 C5            SOUND:  PUSH     BC
 12:    C001 01001C                LD       BC,PSGCOM       ;PSG Register I/O Port
 13:    C004 ED79                  OUT      (C),A           ;Select Register
 14:    C006 05                    DEC      B               ;PSG Data I/O Port
 15:    C007 ED51                  OUT      (C),D           ;Set Data
 16:    C009 C1                    POP      BC
 17:    C00A C9                    RET
 18:
 19:    C00B                       END
```

リスト5-18

```
X1 Self Assembler  Rev 1.0    PAGE     1


  1:                       ;-----------------------------------------------
  2:                       ;       LIST     5-18
  3:                       ;       PSG IN
  4:                       ;       Input    A..Register
  5:                       ;       Output   D..Data
  6:    1C00 =             PSGCOM  EQU       1C00H          ;PSG Register I/O Port
  7:    1B00 =             PSGDAT  EQU       1B00H          ;PSG Data I/O Port
  8:
  9:    C800                       ORG      0C800H
 10:
 11:    C800 C5            PSGIN:  PUSH     BC
 12:    C801 01001C                LD       BC,PSGCOM       ;PSG Register I/O Port
 13:    C804 ED79                  OUT      (C),A           ;Select Register
 14:    C806 05                    DEC      B               ;PSG Data I/O Port
 15:    C807 ED50                  IN       D,(C)           ;Read Data
 16:    C809 C1                    POP      BC
 17:    C80A C9                    RET
 18:
 19:    C80B                       END
```

APPENDIX

付録

付録1　ASCIIコード表(キャラクタ・コード表

下の表から，たとえば大文字のAは＆H41というコードであることがわかります。＆H00～＆H1Fまではコントロール・コード(付録2)です。

付録2　コントロール・コード表

コントロール・コード表

CTRL＋	出力コード	処　理　内　容
@	00	ヌル・コード
A または a	01	INST モードの設定の解除をする。
B　b	02	カーソルを1ワード分左へ戻す。
C　c	03	実行を停止する。
D　d	04	スクリーン・モードなどを初期状態にする。
E　e	05	カーソルから右の行の終わりまで消す。
F　f	06	カーソルを1ワード分右へ進める。
G　g	07	ベルを鳴らす。
H　h	08	文字を抹消する。
I　i	09	水平 TAB (スペース出力)の実行
J　j	0A	ライン・フィード(LF)する。
K　k	0B	ホーム・ポジションへカーソルを移動する。
L　l	0C	画面を消去する。
M　m	0D	キャリッジ・リターン(CR)をする。
N　n	0E	カーソル行から上を上方向へスクロールする。
O　o	0F	カーソル行から下を下方向へスクロールする。
P　p	10	
Q　q	11	一時停止を解除する。
R　r	12	空白を挿入する。
S　s	13	一時停止をする。
T　t	14	水平タブを設定する。
U　u	15	
V　v	16	
W　w	17	次の行と結合する。
X　x	18	
Y　y	19	水平 TAB を抹消する。
Z　z	1A	カーソル以下の画面をすべてクリアする。
[	1B	
¥	1C	カーソルを右へ移動
]	1D	カーソルを左へ移動
∧	1E	カーソルを上へ移動
−	1F	カーソルを下へ移動

付録3　Z80命令表

命令の実行時間はステート数から計算します。X１ではクロックが４MHzなので１ステートの時間は250 n秒（１n秒は１／10^9秒）となります。たとえば、８ビット・ロード命令の"ＬＤ　r，s"はステート数が４なので，４×250n秒＝１μ秒（１μ＝１／10^6）かかります。
フラグの見方は，

・＝影響受けない 0＝リセット 1＝セット ×＝不定 ↕＝実行結果によって変化

となります。

1　8ビット・ロード命令

ニモニック	動　作	フラグ S	Z		H		P/V	N	C	OPコード 76	543	210	16進	バイト数	マシン・サイクル数	ステート数
LD r,s	r←s	・	・	×	・	×	・	・	・	01	r	s		1	1	4
LD r,n	r←n	・	・	×	・	×	・	・	・	00	r	110		2	2	7
										←	n	→				
LD r,(HL)	r←(HL)	・	・	×	・	×	・	・	・	01	r	110		1	2	7
LD r,(IX+d)	r←(IX+d)	・	・	×	・	×	・	・	・	11	011	101	DD	3	5	19
										01	r	110				
										←	d	→				
LD r,(IY+d)	r←(IY+d)	・	・	×	・	×	・	・	・	11	111	101	FD	3	5	19
										01	r	110				
										←	d	→				
LD (HL),r	(HL)←r	・	・	×	・	×	・	・	・	01	110	r		1	2	7
LD (IX+d),r	(IX+d)←r	・	・	×	・	×	・	・	・	11	011	101	DD	3	5	19
										01	110	r				
										←	d	→				
LD (IY+d),r	(IY+d)←r	・	・	×	・	×	・	・	・	11	111	101	FD	3	5	19
										01	110	r				
										←	d	→				
LD (HL),n	(HL)←n	・	・	×	・	×	・	・	・	00	110	110	36	2	3	10
										←	n	→				
LD (IX+d),n	(IX+d)←n	・	・	×	・	×	・	・	・	11	011	101	DD	4	5	19
										00	110	110	36			
										←	d	→				
										←	n	→				
LD (IY+d),n	(IY+d)←n	・	・	×	・	×	・	・	・	11	111	101	FD	4	5	19
										00	110	110	36			
										←	d	→				
										←	n	→				
LD A,(BC)	A←(BC)	・	・	×	・	×	・	・	・	00	001	010	0A	1	2	7
LD A,(DE)	A←(DE)	・	・	×	・	×	・	・	・	00	011	010	1A	1	2	7
LD A,(nn)	A←(nn)	・	・	×	・	×	・	・	・	00	111	010	3A	3	4	13
										←	n	→				
										←	n	→				
LD (BC),A	(BC)←A	・	・	×	・	×	・	・	・	00	000	010	02	1	2	7
LD (DE),A	(DE)←A	・	・	×	・	×	・	・	・	00	010	010	12	1	2	7
LD (nn),A	(nn)←A	・	・	×	・	×	・	・	・	00	110	010	32	3	4	13
										←	n	→				
										←	n	→				
LD A,I	A←I	↕	↕	×	0	×	IFF	0	・	11	101	101	ED	2	2	9
										01	010	111	57			
LD A,R	A←R	↕	↕	×	0	×	IFF	0	・	11	101	101	ED	2	2	9
										01	011	111	5F			
LD I,A	I←A	・	・	×	・	×	・	・	・	11	101	101	ED	2	2	9
										01	000	111	47			
LD R,A	R←A	・	・	×	・	×	・	・	・	11	101	101	ED	2	2	9
										01	001	111	4F			

〈備考〉 r, sはレジスタA, B, C, D, E, H, Lを意味します。

r, s	レジスタ
000	B
001	C
010	D
011	E
100	H
101	L
111	A

IFFは割込み許可フリップ・フロップ(IFF)の内容。

2　16ビット・ロード命令

ニモニック	動　作	フラグ S Z H P/V N C	OPコード 76	 543	 210	 16進	バイト数	マシン・サイクル数	ステート数
LD dd, nn	dd←nn	・・×・×・・・	00	dd0	001		3	3	10
			←	n	→				
			←	n	→				
LD IX, nn	IX←nn	・・×・×・・・	11	011	101	DD	4	4	14
			00	100	001	21			
			←	n	→				
			←	n	→				
LD IY, nn	IY←nn	・・×・×・・・	11	111	101	FD	4	4	14
			00	100	001	21			
			←	n	→				
			←	n	→				
LD HL, (nn)	H←(nn+1)	・・×・×・・・	00	101	010	2A	3	5	16
	L←(nn)	・・×・×・・・	←	n	→				
			←	n	→				
LD dd, (nn)	dd_H←(nn+1)	・・×・×・・・	11	101	101	ED	4	6	20
	dd_L←(nn)	・・×・×・・・	01	dd1	011				
			←	n	→				
			←	n	→				
LD IX, (nn)	IX_H←(nn+1)	・・×・×・・・	11	011	101	DD	4	6	20
	IX_L←(nn)	・・×・×・・・	00	101	010	2A			
			←	n	→				
			←	n	→				
LD IY, (nn)	IY_H←(nn+1)	・・×・×・・・	11	111	101	FD	4	6	20
	IY_L←(nn)	・・×・×・・・	00	101	010	2A			
			←	n	→				
			←	n	→				
LD (nn), HL	(nn+1)←H	・・×・×・・・	00	100	010	22	3	5	16
	(nn)←L	・・×・×・・・	←	n	→				
			←	n	→				
LD (nn), dd	(nn+1)←dd_H	・・×・×・・・	11	101	101	ED	4	6	20
	(nn)←dd_L	・・×・×・・・	01	dd0	011				
			←	n	→				
			←	n	→				
LD (nn), IX	(nn+1)←IX_H	・・×・×・・・	11	011	101	DD	4	6	20
	(nn)←IX_L	・・×・×・・・	00	100	010	22			
			←	n	→				
			←	n	→				
LD (nn), IY	(nn+1)←IY_H	・・×・×・・・	11	111	101	FD	4	6	20
	(nn)←IY_L	・・×・×・・・	00	100	010	22			
			←	n	→				
			←	n	→				
LD SP, HL	SP←HL	・・×・×・・・	11	111	001	F9	1	1	6
LD SP, IX	SP←IX	・・×・×・・・	11	011	101	DD	2	2	10
			11	111	001	F9			
LD SP, IY	SP←IY	・・×・×・・・	11	111	101	FD	2	2	10
			11	111	001	F9			

〈備考〉　ddはペア・レジスタBC, DE, HL, SPを意味します。

dd	ペア
00	BC
01	DE
10	HL
11	SP

$(ペア・レジスタ)_H$, $(ペア・レジスタ)_L$は各ペア・レジスタの上位または下位8ビットを意味します。
　例：BC_L=C,　AF_H=A

3　プッシュ，ポップ命令

ニモニック	動　作	フラグ S Z H P/V N C	OPコード 76	543	210	16進	バイト数	マシン・サイクル数	ステート数
PUSH qq	(SP−2)←qq_L (SP−1)←qq_H	· · × · × · · ·	11	qq0	101		1	3	11
PUSH IX	(SP−2)←IX_L (SP−1)←IX_H	· · × · × · · ·	11 11	011 100	101 101	DD E5	2	4	15
PUSH IY	(SP−2)←IY_L (SP−1)←IY_H	· · × · × · · ·	11 11	111 100	101 101	FD E5	2	4	15
POP qq	qq_H←(SP+1) qq_L←(SP)	· · × · × · · ·	11	qq0	001		1	3	10
POP IX	IX_H←(SP+1) IX_L←(SP)	· · × · × · · ·	11 11	011 100	101 001	DD E1	2	4	14
POP IY	IY_H←(SP+1) IY_L←(SP)	· · × · × · · ·	11 11	111 100	101 001	FD E1	2	4	14

〈備考〉 qqはペア・レジスタAF, BC, DE, HLを意味します。

qq	ペア
00	BC
01	DE
10	HL
11	AF

4　データ交換命令

ニモニック	動　作	フラグ S Z H P/V N C	OPコード 76	543	210	16進	バイト数	マシン・サイクル数	ステート数
EX DE, HL	DE↔HL	· · × · × · · ·	11	101	011	EB	1	1	4
EX AF, AF′	AF↔AF′	· · × · × · · ·	00	001	000	08	1	1	4
EXX	(BC↔BC′ DE↔DE′ HL↔HL′)	· · × · × · · ·	11	011	001	D9	1	1	4
EX (SP), HL	H↔(SP+1) L↔(SP)	· · × · × · · ·	11	100	011	E3	1	5	19
EX (SP), IX	IX_H↔(SP+1) IX_L↔(SP)	· · × · × · · ·	11 11	011 100	101 011	DD E3	2	6	23
EX (SP), IY	IY_H↔(SP+1) IY_L↔(SP)	· · × · × · · ·	11 11	111 100	101 011	FD E3	2	6	23

5　ブロック転送命令

ニモニック	動作	S	Z		H		P/V	N	C	76	543	210	16進	バイト数	マシン・サイクル数	ステート数
LDI	(DE)←(HL) DE←DE＋1 HL←HL＋1 BC←BC－1	·	·	×	0	×	① ↕	0	·	11 10	101 100	101 000	ED A0	2	4	16
LDIR	(DE)←(HL) DE←DE＋1 HL←HL＋1 BC←BC－1 BC=0になるまで	·	·	×	0	×	0	0	·	11 10	101 110	101 000	ED B0	2	(BC≠0のとき) 5 4 (BC=0のとき)	 21 16
LDD	(DE)←(HL) DE←DE－1 HL←HL－1 BC←BC－1	·	·	×	0	×	① ↕	0	·	11 10	101 101	101 000	ED A8	2	4	16
LDDR	(DE)←(HL) DE←DE－1 HL←HL－1 BC←BC－1 BC=0になるまで	·	·	×	0	×	0	0	·	11 10	101 111	101 000	ED B8	2	(BC≠0のとき) 5 4 (BC=0のとき)	 21 16

6　ブロック・サーチ命令

ニモニック	動作	S	Z		H		P/V	N	C	76	543	210	16進	バイト数	マシン・サイクル数	ステート数
CPI	A－(HL) HL←HL＋1 BC←BC－1	↕	② ↕	×	↕	×	① ↕	1	·	11 10	101 100	101 001	ED A1	2	4	16
CPIR	A－(HL) HL←HL＋1 BC←BC－1 A=(HL)または BC=0まで	↕	② ↕	×	↕	×	① ↕	1	·	11 10	101 110	101 001	ED B1	2	5 4	③ 21 16 ④
CPD	A－(HL) HL←HL－1 BC←BC－1	↕	② ↕	×	↕	×	① ↕	1	·	11 10	101 101	101 001	ED A9	2	4	16
CPDR	A－(HL) HL←HL－1 BC←BC－1 A =(HL)または BC=0まで	↕	② ↕	×	↕	×	① ↕	1	·	11 10	101 111	101 001	ED B9	2	5 4	③ 21 16 ④

〈備考〉 ①もしBC－1=0ならばP/V=0，その他P/V=1　④BC=0またはA=(HL)のとき。
②もしA=(HL)ならばZ=1，その他Z=0
③BC≠0かつA≠(HL)のとき

7　8ビット演算命令

ニモニック	動　作	フラグ S Z H P/V N C	OPコード 76	543	210	16進	バイト数	マシン・サイクル数	ステート数
ADD A, r	A←A+r	↕ ↕ × ↕ × V 0 ↕	10	[000]	r		1	1	4
ADD A, n	A←A+n	↕ ↕ × ↕ × V 0 ↕	11	[000]	110		2	2	7
			←	n	→				
ADD A, (HL)	A←A+(HL)	↕ ↕ × ↕ × V 0 ↕	10	[000]	110		1	2	7
ADD A, (IX+d)	A←A+(IX+d)	↕ ↕ × ↕ × V 0 ↕	11	011	101	DD	3	5	19
			10	[000]	110				
			←	d	→				
ADD A, (IY+d)	A←A+(IY+d)	↕ ↕ × ↕ × V 0 ↕	11	111	101	FD	3	5	19
			10	[000]	110				
			←	d	→				
ADC A, s	A←A+s+CY	↕ ↕ × ↕ × V 0 ↕		[001] ①					
SUB s	A←A−s	↕ ↕ × ↕ × V 1 ↕		[010]					
SBC A, s	A←A−s−CY	↕ ↕ × ↕ × V 1 ↕		[011]					
AND s	A←A∧s	↕ ↕ × 1 × P 0 0		[100]					
OR s	A←A∨s	↕ ↕ × 0 × P 0 0		[110]					
XOR s	A←A∀s	↕ ↕ × 0 × P 0 0		[101]					
CP s	A−s	↕ ↕ × ↕ × V 1 ↕		[111]					
INC r	r←r+1	↕ ↕ × ↕ × V 0 ·	00	r	[100]		1	1	4
INC (HL)	(HL)←(HL)+1	↕ ↕ × ↕ × V 0 ·	00	110	[100]		1	3	11
INC (IX+d)	(IX+d)←(IX+d)+1	↕ ↕ × ↕ × V 0 ·	11	011	101	DD	3	6	23
			00	110	[100]				
			←	d	→				
INC (IY+d)	(IY+d)←(IY+d)+1	↕ ↕ × ↕ × V 0 ·	11	111	101	FD	3	6	23
			00	110	[100]				
			←	d	→				
DEC s	s←s−1	↕ ↕ × ↕ × V 1 ·			[101]				
DAA	10進演算補正	↕ ↕ × ↕ × P · ↕	00	100	111	27	1	1	4
CPL	A←$\overline{A}$ (1の補数)	· · × 1 × · 1 ·	00	101	111	2F	1	1	4
NEG	A←$\overline{A}$+1 (2の補数)	↕ ↕ × ↕ × V 1 ↕	11	101	101	ED	2	2	8
			01	000	100	44			

〈備考〉 s=r, n, (HL), (IX+d), (IY+d)のいずれか。

① [000]=[001]～[111]に変わるほかは ADD命令と同様な繰り返し。

INC命令→DEC命令=[100]→[101]に変わるほかは同じ。

r	レジスタ
000	B
001	C
010	D
011	E
100	H
101	L
111	A

P/Vフラグ

論理演算の場合：

偶数パリティ→P=1

奇数パリティ→P=0

算術演算の場合：

オーバーフローあり→V=1

オーバーフローなし→V=0

8　16ビット演算命令

ニモニック	動　作	フラグ S Z H PV N C	OPコード 76	 543	 210	 16進	バイト数	マシン・サイクル数	ステート数
ADD HL, ss	HL←HL+ss	・・×××・0 ↕	00	ss1	001		1	3	11
ADC HL, ss	HL←HL+ss+CY	↕ ↕ ××× V 0 ↕	11	101	101	ED	2	4	15
			01	ss1	010				
SBC HL, ss	HL←HL−ss−CY	↕ ↕ ××× V 1 ↕	11	101	101	ED	2	4	15
			01	ss0	010				
ADD IX, pp	IX←IX+pp	・・×××・0 ↕	11	011	101	DD	2	4	15
			00	pp1	001				
ADD IY, rr	IY←IY+rr	・・×××・0 ↕	11	111	101	FD	2	4	15
			00	rr1	001				
INC ss	ss←ss+1	・・×・×・・・	00	ss0	011		1	1	6
INC IX	IX←IX+1	・・×・×・・・	11	011	101	DD	2	2	10
			00	100	011	23			
INC IY	IY←IY+1	・・×・×・・・	11	111	101	FD	2	2	10
			00	100	011	23			
DEC ss	ss←ss−1	・・×・×・・・	00	ss1	011		1	1	6
DEC IX	IX←IX−1	・・×・×・・・	11	011	101	DD	2	2	10
			00	101	011	2B			
DEC IY	IY←IY−1	・・×・×・・・	11	111	101	ED	2	2	10
			00	101	011	2B			

〈備考〉

ss	レジスタ	pp	レジスタ	rr	レジスタ
00	BC	00	BC	00	BC
01	DE	01	DE	01	DE
10	HL	10	IX	10	IY
11	SP	11	SP	11	SP

9 ローテイト，シフト命令

ニモニック	動作	フラグ S Z H P/V N C	OPコード 76	543	210	16進	バイト数	マシン・サイクル数	ステート数
RLCA	CY ← [7←0] ←（7→0へ循環） A	・ ・ × 0 × ・ 0 ↕	00	000	111	07	1	1	4
RLA	CY ← [7←0] ←（CY→0へ循環） A	・ ・ × 0 × ・ 0 ↕	00	010	111	17	1	1	4
RRCA	[7→0] → CY（0→7へ循環） A	・ ・ × 0 × ・ 0 ↕	00	001	111	0F	1	1	4
RRA	[7→0] → CY（CY→7へ循環） A	・ ・ × 0 × ・ 0 ↕	00	011	111	1F	1	1	4
RLC r	CY ← [7←0] ←（7→0へ循環） r, (HL), (IX+d), (IY+d)	↕ ↕ × 0 × P 0 ↕	11	001	011	CB	2	2	8
			00	[000]	r				
RLC (HL)		↕ ↕ × 0 × P 0 ↕	11	001	011	CB	2	4	15
			00	[000]	110				
RLC (IX+d)		↕ ↕ × 0 × P 0 ↕	11	011	101	DD	4	6	23
			11	001	011	CB			
			←	d	→				
			00	[000]	110				
RLC (IY+d)		↕ ↕ × 0 × P 0 ↕	11	111	101	FD	4	6	23
			11	001	011	CB			
			←	d	→				
			00	[000]	110				
RL s	CY ← [7←0] ←（CY→0へ循環） s≡r, (HL), (IX+d), (IY+d)	↕ ↕ × 0 × P 0 ↕		[010]					
RRC s	[7→0] → CY（0→7へ循環） s≡r, (HL), (IX+d), (IY+d)	↕ ↕ × 0 × P 0 ↕		[001]					
RR s	[7→0] → CY（CY→7へ循環） s≡r, (HL), (IX+d), (IY+d)	↕ ↕ × 0 × P 0 ↕		[011]	①				
SLA s	CY ← [7←0] ← 0 s≡r, (HL), (IX+d), (IY+d)	↕ ↕ × 0 × P 0 ↕		[100]					
SRA s	[7→0] → CY（7→7へ） s≡r, (HL), (IX+d), (IY+d)	↕ ↕ × 0 × P 0 ↕		[101]					
SRL s	0 → [7→0] → CY s≡r, (HL), (IX+d), (IY+d)	↕ ↕ × 0 × P 0 ↕		[111]					
RLD	A [7-4][3-0] [7-4][3-0] (HL)	↕ ↕ × 0 × P 0 ・	11	101	101	ED	2	5	18
			01	101	111	6F			
RRD	A [7-4][3-0] [7-4][3-0] (HL)	↕ ↕ × 0 × P 0 ・	11	101	101	ED	2	5	18
			01	100	111	67			

〈備考〉

r	レジスタ
000	B
001	C
010	D
011	E
100	H
101	L
111	A

① [000]=[001]～[111]に変わるほかは，RLC命令と同様な繰り返し。

10　ビット操作命令

ニモニック	動　作	フ　ラ　グ	OPコード				バイト数	マシン・サイクル数	ステート数
		S Z H P/V N C	76	543	210	16進			
BIT b, r	$Z \leftarrow \overline{r_b}$	× ↕ × 1 × × 0 ·	11	001	011	CB	2	2	8
			01	b	r				
BIT b, (HL)	$Z \leftarrow \overline{(HL)_b}$	× ↕ × 1 × × 0 ·	11	001	011	CB	2	3	12
			01	b	110				
BIT b, (IX+d)	$Z \leftarrow \overline{(IX+d)_b}$	× ↕ × 1 × × 0 ·	11	011	101	DD	4	5	20
			11	001	011	CB			
			←	d	→				
			01	b	110				
BIT b, (IY+d)	$Z \leftarrow (IY+d)_b$	× ↕ × 1 × × 0 ·	11	111	101	FD	4	5	20
			11	001	011	CB			
			←	d	→				
			01	b	110				
SET b, r	$r_b \leftarrow 1$	· · × · × · · ·	11	001	011	CB	2	2	8
			[11]	b	r				
SET b, (HL)	$(HL)_b \leftarrow 1$	· · × · × · · ·	11	001	011	CB	2	4	15
			[11]	b	110				
SET b, (IX+d)	$(IX+d)_b \leftarrow 1$	· · × · × · · ·	11	011	101	DD	4	6	23
			11	001	011	CB			
			←	d	→				
			[11]	b	110				
SET b, (IY+d)	$(IY+d)_b \leftarrow 1$	· · × · × · · ·	11	111	101	FD	4	6	23
			11	001	011	CB			
			←	d	→				
			[11]	b	110				
RES b, s	$s_b \leftarrow 0$ $s \equiv r, (HL), (IX+d), (IY+d)$		[10]						

〈備考〉

r	レジスタ
000	B
001	C
010	D
011	E
100	H
101	L
111	A

b	Bit
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

SET命令→RES命令＝[11]→[10]に変えて同様な繰り返し

11　フラグ操作命令

ニモニック	動　作	フ　ラ　グ	OPフラグ				バイト数	マシン・サイクル数	ステート数
		S Z H P/V N C	76	543	210	16進			
CCF	$CY \leftarrow \overline{CY}$	· · × × × · 0 ↕	00	111	111	3F	1	1	4
SCF	$CY \leftarrow 1$	· · × 0 × · 0 1	00	110	111	37	1	1	4

12　ジャンプ命令

ニモニック	動作	フラグ S Z H P/V N C	OPコード 76	 543	 210	 16進	バイト数	マシン・サイクル数	ステート数
JP nn	PC←nn	・ ・ × ・ × ・ ・ ・	11	000	011	C3	3	3	10
			←	n	→				
			←	n	→				
JP cc,nn	もしccが成り立てばジャンプ。そうでなければ次命令へ。	・ ・ × ・ × ・ ・ ・	11	cc	010		3	3	10
			←	n	→				
			←	n	→				
JR e	PC←PC+e	・ ・ × ・ × ・ ・ ・	00	011	000	18	2	3	12
			←	e−2	→				
JR C,e	C=0なら次命令へ。	・ ・ × ・ × ・ ・ ・	00	111	000	38	2	2	7
	C=1なら PC←PC+e		←	e−2	→		2	3	12
JR NC,e	C=1なら次命令へ。	・ ・ × ・ × ・ ・ ・	00	110	000	30	2	2	7
	C=0なら PC←PC+e		←	e−2	→		2	3	12
JR Z,e	Z=0なら次命令へ。	・ ・ × ・ × ・ ・ ・	00	101	000	28	2	2	7
	Z=1なら PC←PC+e		←	e−2	→		2	3	12
JR NZ,e	Z=1なら次命令へ。	・ ・ × ・ × ・ ・ ・	00	100	000	20	2	2	7
	Z=0なら PC←PC+e		←	e−2	→		2	3	12
JP(HL)	PC←HL	・ ・ × ・ × ・ ・ ・	11	101	001	E9	1	1	4
JP(IX)	PC←IX	・ ・ × ・ × ・ ・ ・	11	011	101	DD	2	2	8
			11	101	001	E9			
JP(IY)	PC←IY	・ ・ × ・ × ・ ・ ・	11	111	101	FD	2	2	8
			11	101	001	E9			
DJNZ e	B←B−1	・ ・ × ・ × ・ ・ ・	00	010	000	10	2	2	8
	B=0なら次命令へ。 B≠0ならPC←PC+e		←	e−2	→		2	3	13

〈備考〉 e=レラティブ・アトレッシング・モードにおける変位値
（符号付き2の補数=+127～−128）
e−2=eの実効変位値

cc	条件
000	NZ
001	Z
010	NC
011	C
100	PO
101	PE
110	P
111	M

13 コール，リターン命令

ニモニック	動　作	フラグ	OPコード				バイト数	マシン・サイクル数	ステート数
		S Z H P/V N C	76	543	210	16進			
CALL nn	(SP−1)←PC_H	• • × • × • • •	11	001	101	CD	3	5	17
	(SP−2)←PC_L		←	n	→				
	PC←nn		←	n	→				
CALL cc,nn	もしccが成り立てばコール。そうでなければ次命令へ。	• • × • × • • •	11	cc	100		3	3	10
			←	n	→				
			←	n	→		3	5	17
RET	PC_L←(SP)	• • × • × • • •	11	001	001	C9	1	3	10
	PC_H←(SP+1)								
RET cc	もしccが成り立てばリターン。そうでなければ次命令へ。	• • × • × • • •	11	cc	000		1	1	5
							1	3	11
RETI	インタラプトから戻る。	• • × • × • • •	11	101	101	ED	2	4	14
			01	001	101	4D			
RETN	ノンマスカブル・インタラプトから戻る。	• • × • × • • •	11	101	101	ED	2	4	14
			01	000	101	45			
RST p	(SP−1)←PC_H	• • × • × • • •	11	t	111		1	3	11
	(SP−2)←PC_L								
	PC_H←0								
	PC_L←p								

〈備考〉

cc	条件
000	NZ
001	Z
010	NC
011	C
100	PO
101	PE
110	P
111	M

t	p
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

14　入出力命令

ニモニック	動作	フラグ S Z H PV N C	OPコード 76	543	210	16進	バイト数	マシン・サイクル数	ステート数
IN A,n	A←(n)	• • × • × • • •	11	011	011	DB	2	3	11
			←	n	→				
IN r,(C)	r←(C)	↕ ↕ × 0 × P 0 •	11	101	101	ED	2	3	12
			01	r	000				
		①							
INI	(HL)←(C)	× ↕ × × × × 1 •	11	101	101	ED	2	4	16
	B←B−1		10	100	010	A2			
	HL←HL+1								
INIR	(HL)←(C)	× 1 × × × × 1 •	11	101	101	ED	2	5	21
	B←B−1		10	110	010	B2		(If B≠0)	
	HL←HL+1						2	4	16
	B=0になるまでく							(If B=0)	
	りかえす。								
		①							
IND	(HL)←(C)	× ↕ × × × × 1 •	11	101	101	ED	2	4	16
	B←B−1		10	101	010	AA			
	HL←HL−1								
INDR	(HL)←(C)	× 1 × × × × 1 •	11	101	101	ED	2	5	21
	B←B−1		10	111	010	BA		(If B≠0)	
	HL←HL−1						2	4	16
	B=0 になるまでく							(If B=0)	
	りかえす。								
OUT n,A	(n)←A	• • × • × • • •	11	010	011	D3	2	3	11
			←	n	→				
OUT(C),r	(C)←r	• • × • × • • •	11	101	101	ED	2	3	12
			01	r	001				
		①							
OUTI	(C)←(HL)	× ↕ × × × × 1 •	11	101	101	ED	2	4	16
	B←B−1		10	100	011	A3			
	HL←HL+1								
OTIR	(C)←(HL)	× 1 × × × × 1 •	11	101	101	ED	2	5	21
	B←B−1		10	110	011	B3		(If B≠0)	
	HL←HL+1						2	4	16
	B=0 になるまでく							(If B=0)	
	りかえす。								
		①							
OUTD	(C)←(HL)	× ↕ × × × × 1 •	11	101	101	ED	2	4	16
	B←B−1		10	101	011	AB			
	HL←HL−1								
OTDR	(C)←(HL)	× 1 × × × × 1 •	11	101	101	ED	2	5	21
	B←B−1		10	111	011	BB		(If B≠0)	
	HL←HL−1						2	4	16
	B=0 になるまでく							(If B=0)	
	りかえす。								

〈備考〉 ①もしB−1=0ならばZ=1，その他Z=0　rは10.ビット操作命令の備考を参照

15 CPUコントロール命令

ニモニック	動作	フラグ S Z H P/V N C	OPコード 76	 543	 210	 16進	バイト数	マシン・サイクル数	ステート数
NOP	No operation	・ ・ × ・ × ・ ・ ・	00	000	000	00	1	1	4
HALT	CPU停止	・ ・ × ・ × ・ ・ ・	01	110	110	76	1	1	4
DI	IFF←0	・ ・ × ・ × ・ ・ ・	11	110	011	F3	1	1	4
EI	IFF←1	・ ・ × ・ × ・ ・ ・	11	111	011	FB	1	1	4
IM 0	割込モード0	・ ・ × ・ × ・ ・ ・	11 01	101 000	101 110	FD 46	2	2	8
IM 1	割込モード1	・ ・ × ・ × ・ ・ ・	11 01	101 010	101 110	ED 56	2	2	8
IM 2	割込モード2	・ ・ × ・ × ・ ・ ・	11 01	101 011	101 110	ED 5E	2	2	8

〈参考〉 IFF＝割込許可フリップ・フロップ

付録4 マシン語↔ニモニック対応表

マシン語 ⟷ ニモニック											
00	NOP		40	LD	B,B	80	ADD	A,B	C0	RET	NZ
01	LD	BC,nn	41	LD	B,C	81	ADD	A,C	C1	POP	BC
02	LD	(BC),A	42	LD	B,D	82	ADD	A,D	C2	JP	NZ,nn
03	INC	BC	43	LD	B,E	83	ADD	A,E	C3	JP	nn
04	INC	B	44	LD	B,H	84	ADD	A,H	C4	CALL	NZ,nn
05	DEC	B	45	LD	B,L	85	ADD	A,L	C5	PUSH	BC
06	LD	B,n	46	LD	B,(HL)	86	ADD	A,(HL)	C6	ADD	A,n
07	RLCA		47	LD	B,A	87	ADD	A,A	C7	RST	00H
08	EX	AF,AF	48	LD	C,B	88	ADC	A,B	C8	RET	Z
09	ADD	HL,BC	49	LD	C,C	89	ADC	A,C	C9	RET	
0A	LD	A,(BC)	4A	LD	C,D	8A	ADC	A,D	CA	JP	Z,nn
0B	DEC	BC	4B	LD	C,E	8B	ADC	A,E	CB		
0C	INC	C	4C	LD	C,H	8C	ADC	A,H	CC	CALL	Z,nn
0D	DEC	C	4D	LD	C,L	8D	ADC	A,L	CD	CALL	nn
0E	LD	C,n	4E	LD	C,(HL)	8E	ADC	A,(HL)	CE	ADC	A,n
0F	RRCA		4F	LD	C,A	8F	ADC	A,A	CF	RST	08H
10	DJNZ	e	50	LD	D,B	90	SUB	B	D0	RET	NC
11	LD	DE,nn	51	LD	D,C	91	SUB	C	D1	POP	DE
12	LD	(DE),A	52	LD	D,D	92	SUB	D	D2	JP	NC,nn
13	INC	DE	53	LD	D,E	93	SUB	E	D3	OUT	n,A
14	INC	D	54	LD	D,H	94	SUB	H	D4	CALL	NC,nn
15	DEC	D	55	LD	D,L	95	SUB	L	D5	PUSH	DE
16	LD	D,n	56	LD	D,(HL)	96	SUB	(HL)	D6	SUB	n
17	RLA		57	LD	D,A	97	SUB	A	D7	RST	10H
18	JR	e	58	LD	E,B	98	SBC	A,B	D8	RET	C
19	ADD	HL,DE	59	LD	E,C	99	SBC	A,C	D9	EXX	
1A	LD	A,(DE)	5A	LD	E,D	9A	SBC	A,D	DA	JP	C,nn
1B	DEC	DE	5B	LD	E,E	9B	SBC	A,E	DB	IN	A,n
1C	INC	E	5C	LD	E,H	9C	SBC	A,H	DC	CALL	C,nn
1D	DEC	E	5D	LD	E,L	9D	SBC	A,L	DD		
1E	LD	E,n	5E	LD	E,(HL)	9E	SBC	A,(HL)	DE	SBC	A,n
1F	RRA		5F	LD	E,A	9F	SBC	A,A	DF	RST	18H
20	JR	NZ,e	60	LD	H,B	A0	AND	B	E0	RET	PO
21	LD	HL,nn	61	LD	H,C	A1	AND	C	E1	POP	HL
22	LD	(nn),HL	62	LD	H,D	A2	AND	D	E2	JP	PO,nn
23	INC	HL	63	LD	H,E	A3	AND	E	E3	EX	(SP),HL
24	INC	H	64	LD	H,H	A4	AND	H	E4	CALL	PO,nn
25	DEC	H	65	LD	H,L	A5	AND	L	E5	PUSH	HL
26	LD	H,n	66	LD	H,(HL)	A6	AND	(HL)	E6	AND	n
27	DAA		67	LD	H,A	A7	AND	A	E7	RST	20H
28	JR	Z,e	68	LD	L,B	A8	XOR	B	E8	RET	PE
29	ADD	HL,HL	69	LD	L,C	A9	XOR	C	E9	JP	(HL)
2A	LD	HL,(nn)	6A	LD	L,D	AA	XOR	D	EA	JP	PE,nn
2B	DEC	HL	6B	LD	L,E	AB	XOR	E	EB	EX	DE,HL
2C	INC	L	6C	LD	L,H	AC	XOR	H	EC	CALL	PE,nn
2D	DEC	L	6D	LD	L,L	AD	XOR	L	ED		
2E	LD	L,n	6E	LD	L,(HL)	AE	XOR	(HL)	EE	XOR	n
2F	CPL		6F	LD	L,A	AF	XOR	A	EF	RST	28H
30	JR	NC,e	70	LD	(HL),B	B0	OR	B	F0	RET	P
31	LD	SP,nn	71	LD	(HL),C	B1	OR	C	F1	POP	AF
32	LD	(nn),A	72	LD	(HL),D	B2	OR	D	F2	JP	P.nn
33	INC	SP	73	LD	(HL),E	B3	OR	E	F3	DI	
34	INC	(HL)	74	LD	(HL),H	B4	OR	H	F4	CALL	P,nn
35	DEC	(HL)	75	LD	(HL),L	B5	OR	L	F5	PUSH	AF
36	LD	(HL),n	76	HALT		B6	OR	(HL)	F6	OR	n
37	SCF		77	LD	(HL),A	B7	OR	A	F7	RST	30H
38	JR	C,e	78	LD	A,B	B8	CP	B	F8	RET	M
39	ADD	HL,SP	79	LD	A,C	B9	CP	C	F9	LD	SP,HL
3A	LD	A,(nn)	7A	LD	A,D	BA	CP	D	FA	JP	M,nn
3B	DEC	SP	7B	LD	A,E	BB	CP	E	FB	EI	
3C	INC	A	7C	LD	A,H	BC	CP	H	FC	CALL	M,nn
3D	DEC	A	7D	LD	A,L	BD	CP	L	FD		
3E	LD	A,n	7E	LD	A,(HL)	BE	CP	(HL)	FE	CP	n
3F	CCF		7F	LD	A,A	BF	CP	A	FF	RST	38H

CB XX

Code	Instr.	Operand	Code	Instr.	Operand	Code	Instr.	Operand	Code	Instr.	Operand
00	RLC	B	40	BIT	0, B	80	RES	0, B	C0	SET	0, B
01	RLC	C	41	BIT	0, C	81	RES	0, C	C1	SET	0, C
02	RLC	D	42	BIT	0, D	82	RES	0, D	C2	SET	0, D
03	RLC	E	43	BIT	0, E	83	RES	0, E	C3	SET	0, E
04	RLC	H	44	BIT	0, H	84	RES	0, H	C4	SET	0, H
05	RLC	L	45	BIT	0, L	85	RES	0, L	C5	SET	0, L
06	RLC	(HL)	46	BIT	0, (HL)	86	RES	0, (HL)	C6	SET	0, (HL)
07	RLC	A	47	BIT	0, A	87	RES	0, A	C7	SET	0, A
08	RRC	B	48	BIT	1, B	88	RES	1, B	C8	SET	1, B
09	RRC	C	49	BIT	1, C	89	RES	1, C	C9	SET	1, C
0A	RRC	D	4A	BIT	1, D	8A	RES	1, D	CA	SET	1, D
0B	RRC	E	4B	BIT	1, E	8B	RES	1, E	CB	SET	1, E
0C	RRC	H	4C	BIT	1, H	8C	RES	1, H	CC	SET	1, H
0D	RRC	L	4D	BIT	1, L	8D	RES	1, L	CD	SET	1, L
0E	RRC	(HL)	4E	BIT	1, (HL)	8E	RES	1, (HL)	CE	SET	1, (HL)
0F	RRC	A	4F	BIT	1, A	8F	RES	1, A	CF	SET	1, A
10	RL	B	50	BIT	2, B	90	RES	2, B	D0	SET	2, B
11	RL	C	51	BIT	2, C	91	RES	2, C	D1	SET	2, C
12	RL	D	52	BIT	2, D	92	RES	2, D	D2	SET	2, D
13	RL	E	53	BIT	2, E	93	RES	2, E	D3	SET	2, E
14	RL	H	54	BIT	2, H	94	RES	2, H	D4	SET	2, H
15	RL	L	55	BIT	2, L	95	RES	2, L	D5	SET	2, L
16	RL	(HL)	56	BIT	2, (HL)	96	RES	2, (HL)	D6	SET	2, (HL)
17	RL	A	57	BIT	2, A	97	RES	2, A	D7	SET	2, A
18	RR	B	58	BIT	3, B	98	RES	3, B	D8	SET	3, B
19	RR	C	59	BIT	3, C	99	RES	3, C	D9	SET	3, C
1A	RR	D	5A	BIT	3, D	9A	RES	3, D	DA	SET	3, D
1B	RR	E	5B	BIT	3, E	9B	RES	3, E	DB	SET	3, E
1C	RR	H	5C	BIT	3, H	9C	RES	3, H	DC	SET	3, H
1D	RR	L	5D	BIT	3, L	9D	RES	3, L	DD	SET	3, L
1E	RR	(HL)	5E	BIT	3, (HL)	9E	RES	3, (HL)	DE	SET	3, (HL)
1F	RR	A	5F	BIT	3, A	9F	RES	3, A	DF	SET	3, A
20	SLA	B	60	BIT	4, B	A0	RES	4, B	E0	SET	4, B
21	SLA	C	61	BIT	4, C	A1	RES	4, C	E1	SET	4, C
22	SLA	D	62	BIT	4, D	A2	RES	4, D	E2	SET	4, D
23	SLA	E	63	BIT	4, E	A3	RES	4, E	E3	SET	4, E
24	SLA	H	64	BIT	4, H	A4	RES	4, H	E4	SET	4, H
25	SLA	L	65	BIT	4, L	A5	RES	4, L	E5	SET	4, L
26	SLA	(HL)	66	BIT	4, (HL)	A6	RES	4, (HL)	E6	SET	4, (HL)
27	SLA	A	67	BIT	4, A	A7	RES	4, A	E7	SET	4, A
28	SRA	B	68	BIT	5, B	A8	RES	5, B	E8	SET	5, B
29	SRA	C	69	BIT	5, C	A9	RES	5, C	E9	SET	5, C
2A	SRA	D	6A	BIT	5, D	AA	RES	5, D	EA	SET	5, D
2B	SRA	E	6B	BIT	5, E	AB	RES	5, E	EB	SET	5, E
2C	SRA	H	6C	BIT	5, H	AC	RES	5, H	EC	SET	5, H
2D	SRA	L	6D	BIT	5, L	AD	RES	5, L	ED	SET	5, L
2E	SRA	(HL)	6E	BIT	5, (HL)	AE	RES	5, (HL)	EE	SET	5, (HL)
2F	SRA	A	6F	BIT	5, A	AF	RES	5, A	EF	SET	5, A
30			70	BIT	6, B	B0	RES	6, B	F0	SET	6, B
31			71	BIT	6, C	B1	RES	6, C	F1	SET	6, C
32			72	BIT	6, D	B2	RES	6, D	F2	SET	6, D
33			73	BIT	6, E	B3	RES	6, E	F3	SET	6, E
34			74	BIT	6, H	B4	RES	6, H	F4	SET	6, H
35			75	BIT	6, L	B5	RES	6, L	F5	SET	6, L
36			76	BIT	6, (HL)	B6	RES	6, (HL)	F6	SET	6, (HL)
37			77	BIT	6, A	B7	RES	6, A	F7	SET	6, A
38	SRL	B	78	BIT	7, B	B8	RES	7, B	F8	SET	7, B
39	SRL	C	79	BIT	7, C	B9	RES	7, C	F9	SET	7, C
3A	SRL	D	7A	BIT	7, D	BA	RES	7, D	FA	SET	7, D
3B	SRL	E	7B	BIT	7, E	BB	RES	7, E	FB	SET	7, E
3C	SRL	H	7C	BIT	7, H	BC	RES	7, H	FC	SET	7, H
3D	SRL	L	7D	BIT	7, L	BD	RES	7, L	FD	SET	7, L
3E	SRL	(HL)	7E	BIT	7, (HL)	BE	RES	7, (HL)	FE	SET	7, (HL)
3F	SRL	A	7F	BIT	7, A	BF	RES	7, A	FF	SET	7, A

DD XX

Code	Mnemonic	Operand	Code	Mnemonic	Operand	Code	Mnemonic	Operand	Code	Mnemonic	Operand
00			40			80			C0		
01			41			81			C1		
02			42			82			C2		
03			43			83			C3		
04			44			84			C4		
05			45			85			C5		
06			46	LD	B, (IX+d)	86	ADD	A, (IX+d)	C6		
07			47			87			C7		
08			48			88			C8		
09	ADD	IX, BC	49			89			C9		
0A			4A			8A			CA		
0B			4B			8B			CB		
0C			4C			8C			CC		
0D			4D			8D			CD		
0E			4E	LD	C, (IX+d)	8E	ADC	A, (IX+d)	CE		
0F			4F			8F			CF		
10			50			90			D0		
11			51			91			D1		
12			52			92			D2		
13			53			93			D3		
14			54			94			D4		
15			55			95			D5		
16			56	LD	D, (IX+d)	96	SUB	(IX+d)	D6		
17			57			97			D7		
18			58			98			D8		
19	ADD	IX, DE	59			99			D9		
1A			5A			9A			DA		
1B			5B			9B			DB		
1C			5C			9C			DC		
1D			5D			9D			DD		
1E			5E	LD	E, (IX+d)	9E	SBC	A, (IX+d)	DE		
1F			5F			9F			DF		
20			60			A0			E0		
21	LD	IX, nn	61			A1			E1	POP	IX
22	LD	(nn), IX	62			A2			E2		
23	INC	IX	63			A3			E3	EX	(SP), IX
24			64			A4			E4		
25			65			A5			E5	PUSH	IX
26			66	LD	H, (IX+d)	A6	AND	(IX+d)	E6		
27			67			A7			E7		
28			68			A8			E8		
29	ADD	IX, IX	69			A9			E9	JP	(IX)
2A	LD	IX, (nn)	6A			AA			EA		
2B	DEC	IX	6B			AB			EB		
2C			6C			AC			EC		
2D			6D			AD			ED		
2E			6E	LD	L, (IX+d)	AE	XOR	(IX+d)	EE		
2F			6F			AF			EF		
30			70	LD	(IX+d), B	B0			F0		
31			71	LD	(IX+d), C	B1			F1		
32			72	LD	(IX+d), D	B2			F2		
33			73	LD	(IX+d), E	B3			F3		
34	INC	(IX+d)	74	LD	(IX+d), H	B4			F4		
35	DEC	(IX+d)	75	LD	(IX+d), L	B5			F5		
36	LD	(IX+d), n	76			B6	OR	(IX+d)	F6		
37			77	LD	(IX+d), A	B7			F7		
38			78			B8			F8		
39	ADD	IX, SP	79			B9			F9	LD	SP, IX
3A			7A			BA			FA		
3B			7B			BB			FB		
3C			7C			BC			FC		
3D			7D			BD			FD		
3E			7E	LD	A, (IX+d)	BE	CP	(IX+d)	FE		
3F			7F			BF			FF		

DD CB —d— XX

Code	Mnemonic	Operand	Code	Mnemonic	Operand	Code	Mnemonic	Operand	Code	Mnemonic	Operand
00			40			80			C0		
01			41			81			C1		
02			42			82			C2		
03			43			83			C3		
04			44			84			C4		
05			45			85			C5		
06	RLC	(IX+d)	46	BIT	0,(IX+d)	86	RES	0,(IX+d)	C6	SET	0,(IX+d)
07			47			87			C7		
08			48			88			C8		
09			49			89			C9		
0A			4A			8A			CA		
0B			4B			8B			CB		
0C			4C			8C			CC		
0D			4D			8D			CD		
0E	RRC	(IX+d)	4E	BIT	1,(IX+d)	8E	RES	1,(IX+d)	CE	SET	1,(IX+d)
0F			4F			8F			CF		
10			50			90			D0		
11			51			91			D1		
12			52			92			D2		
13			53			93			D3		
14			54			94			D4		
15			55			95			D5		
16	RL	(IX+d)	56	BIT	2,(IX+d)	96	RES	2,(IX+d)	D6	SET	2,(IX+d)
17			57			97			D7		
18			58			98			D8		
19			59			99			D9		
1A			5A			9A			DA		
1B			5B			9B			DB		
1C			5C			9C			DC		
1D			5D			9D			DD		
1E	RR	(IX+d)	5E	BIT	3,(IX+d)	9E	RES	3,(IX+d)	DE	SET	3,(IX+d)
1F			5F			9F			DF		
20			60			A0			E0		
21			61			A1			E1		
22			62			A2			E2		
23			63			A3			E3		
24			64			A4			E4		
25			65			A5			E5		
26	SLA	(IX+d)	66	BIT	4,(IX+d)	A6	RES	4,(IX+d)	E6	SET	4,(IX+d)
27			67			A7			E7		
28			68			A8			E8		
29			69			A9			E9		
2A			6A			AA			EA		
2B			6B			AB			EB		
2C			6C			AC			EC		
2D			6D			AD			ED		
2E	SRA	(IX+d)	6E	BIT	5,(IX+d)	AE	RES	5,(IX+d)	EE	SET	5,(IX+d)
2F			6F			AF			EF		
30			70			B0			F0		
31			71			B1			F1		
32			72			B2			F2		
33			73			B3			F3		
34			74			B4			F4		
35			75			B5			F5		
36			76	BIT	6,(IX+d)	B6	RES	6,(IX+d)	F6	SET	6,(IX+d)
37			77			B7			F7		
38			78			B8			F8		
39			79			B9			F9		
3A			7A			BA			FA		
3B			7B			BB			FB		
3C			7C			BC			FC		
3D			7D			BD			FD		
3E	SRL	(IX+d)	7E	BIT	7,(IX+d)	BE	RES	7,(IX+d)	FE	SET	7,(IX+d)
3F			7F			BF			FF		

ED XX			
00	40 IN B,(C)	80	C0
01	41 OUT (C),B	81	C1
02	42 SBC HL,BC	82	C2
03	43 LD (nn),BC	83	C3
04	44 NEG	84	C4
05	45 RETN	85	C5
06	46 IM 0	86	C6
07	47 LD I,A	87	C7
08	48 IN C,(C)	88	C8
09	49 OUT (C),C	89	C9
0A	4A ADC HL,BC	8A	CA
0B	4B LD BC,(nn)	8B	CB
0C	4C	8C	CC
0D	4D RETI	8D	CD
0E	4E	8E	CE
0F	4F LD R,A	8F	CF
10	50 IN D,(C)	90	D0
11	51 OUT (C),D	91	D1
12	52 SBC HL,DE	92	D2
13	53 LD (nn),DE	93	D3
14	54	94	D4
15	55	95	D5
16	56 IM 1	96	D6
17	57 LD A,I	97	D7
18	58 IN E,(C)	98	D8
19	59 OUT (C),E	99	D9
1A	5A ADC HL,DE	9A	DA
1B	5B LD DE,(nn)	9B	DB
1C	5C	9C	DC
1D	5D	9D	DD
1E	5E IM 2	9E	DE
1F	5F LD A,R	9F	DF
20	60 IN H,(C)	A0 LDI	E0
21	61 OUT (C),H	A1 CPI	E1
22	62 SBC HL,HL	A2 INI	E2
23	63	A3 OUTI	E3
24	64	A4	E4
25	65	A5	E5
26	66	A6	E6
27	67 RRD	A7	E7
28	68 IN L,(C)	A8 LDD	E8
29	69 OUT (C),L	A9 CPD	E9
2A	6A ADC HL,HL	AA IND	EA
2B	6B	AB OUTD	EB
2C	6C	AC	EC
2D	6D	AD	ED
2E	6E	AE	EE
2F	6F RLD	AF	EF
30	70	B0 LDIR	F0
31	71	B1 CPIR	F1
32	72 SBC HL,SP	B2 INIR	F2
33	73 LD (nn),SP	B3 OTIR	F3
34	74	B4	F4
35	75	B5	F5
36	76	B6	F6
37	77	B7	F7
38	78 IN A,(C)	B8 LDDR	F8
39	79 OUT (C),A	B9 CPDR	F9
3A	7A ADC HL,SP	BA INDR	FA
3B	7B LD SP,(nn)	BB OTDR	FB
3C	7C	BC	FC
3D	7D	BD	FD
3E	7E	BE	FE
3F	7F	BF	FF

FD XX

Code	Mnemonic	Operand	Code	Mnemonic	Operand	Code	Mnemonic	Operand	Code	Mnemonic	Operand
00			40			80			C0		
01			41			81			C1		
02			42			82			C2		
03			43			83			C3		
04			44			84			C4		
05			45			85			C5		
06			46	LD	B, (IY+d)	86	ADD	A, (IY+d)	C6		
07			47			87			C7		
08			48			88			C8		
09	ADD	IY, BC	49			89			C9		
0A			4A			8A			CA		
0B			4B			8B			CB		
0C			4C			8C			CC		
0D			4D			8D			CD		
0E			4E	LD	C, (IY+d)	8E	ADC	A, (IY+d)	CE		
0F			4F			8F			CF		
10			50			90			D0		
11			51			91			D1		
12			52			92			D2		
13			53			93			D3		
14			54			94			D4		
15			55			95			D5		
16			56	LD	D, (IY+d)	96	SUB	(IY+d)	D6		
17			57			97			D7		
18			58			98			D8		
19	ADD	IY, DE	59			99			D9		
1A			5A			9A			DA		
1B			5B			9B			DB		
1C			5C			9C			DC		
1D			5D			9D			DD		
1E			5E	LD	E, (IY+d)	9E	SBC	A, (IY+d)	DE		
1F			5F			9F			DF		
20			60			A0			E0		
21	LD	IY, nn	61			A1			E1	POP	IY
22	LD	(nn), IY	62			A2			E2		
23	INC	IY	63			A3			E3	EX	(SP), IY
24			64			A4			E4		
25			65			A5			E5	PUSH	IY
26			66	LD	H, (IY+d)	A6	AND	(IY+d)	E6		
27			67			A7			E7		
28			68			A8			E8		
29	ADD	IY, IY	69			A9			E9	JP	(IY)
2A	LD	IY, (nn)	6A			AA			EA		
2B	DEC	IY	6B			AB			EB		
2C			6C			AC			EC		
2D			6D			AD			ED		
2E			6E	LD	L, (IY+d)	AE	XOR	(IY+d)	EE		
2F			6F			AF			EF		
30			70	LD	(IY+d), B	B0			F0		
31			71	LD	(IY+d), C	B1			F1		
32			72	LD	(IY+d), D	B2			F2		
33			73	LD	(IY+d), E	B3			F3		
34	INC	(IY+d)	74	LD	(IY+d), H	B4			F4		
35	DEC	(IY+d)	75	LD	(IY+d), L	B5			F5		
36	LD	(IY+d), n	76			B6	OR	(IY+d)	F6		
37			77	LD	(IY+d), A	B7			F7		
38			78			B8			F8		
39	ADD	IY, SP	79			B9			F9	LD	SP, IY
3A			7A			BA			FA		
3B			7B			BB			FB		
3C			7C			BC			FC		
3D			7D			BD			FD		
3E			7E	LD	A, (IY+d)	BE	CP	(IY+d)	FE		
3F			7F			BF			FF		

FD CB —d— XX

Code	Instr.	Operand	Code	Instr.	Operand	Code	Instr.	Operand	Code	Instr.	Operand
00			40			80			C0		
01			41			81			C1		
02			42			82			C2		
03			43			83			C3		
04			44			84			C4		
05			45			85			C5		
06	RLC	(IY+d)	46	BIT	0,(IY+d)	86	RES	0,(IY+d)	C6	SET	0,(IY+d)
07			47			87			C7		
08			48			88			C8		
09			49			89			C9		
0A			4A			8A			CA		
0B			4B			8B			CB		
0C			4C			8C			CC		
0D			4D			8D			CD		
0E	RRC	(IY+d)	4E	BIT	1,(IY+d)	8E	RES	1,(IY+d)	CE	SET	1,(IY+d)
0F			4F			8F			CF		
10			50			90			D0		
11			51			91			D1		
12			52			92			D2		
13			53			93			D3		
14			54			94			D4		
15			55			95			D5		
16	RL	(IY+d)	56	BIT	2,(IY+d)	96	RES	2,(IY+d)	D6	SET	2,(IY+d)
17			57			97			D7		
18			58			98			D8		
19			59			99			D9		
1A			5A			9A			DA		
1B			5B			9B			DB		
1C			5C			9C			DC		
1D			5D			9D			DD		
1E	RR	(IY+d)	5E	BIT	3,(IY+d)	9E	RES	3,(IY+d)	DE	SET	3,(IY+d)
1F			5F			9F			DF		
20			60			A0			E0		
21			61			A1			E1		
22			62			A2			E2		
23			63			A3			E3		
24			64			A4			E4		
25			65			A5			E5		
26	SLA	(IY+d)	66	BIT	4,(IY+d)	A6	RES	4,(IY+d)	E6	SET	4,(IY+d)
27			67			A7			E7		
28			68			A8			E8		
29			69			A9			E9		
2A			6A			AA			EA		
2B			6B			AB			EB		
2C			6C			AC			EC		
2D			6D			AD			ED		
2E	SRA	(IY+d)	6E	BIT	5,(IY+d)	AE	RES	5,(IY+d)	EE	SET	5,(IY+d)
2F			6F			AF			EF		
30			70			B0			F0		
31			71			B1			F1		
32			72			B2			F2		
33			73			B3			F3		
34			74			B4			F4		
35			75			B5			F5		
36			76	BIT	6,(IY+d)	B6	RES	6,(IY+d)	F6	SET	6,(IY+d)
37			77			B7			F7		
38			78			B8			F8		
39			79			B9			F9		
3A			7A			BA			FA		
3B			7B			BB			FB		
3C			7C			BC			FC		
3D			7D			BD			FD		
3E	SRL	(IY+d)	7E	BIT	7,(IY+d)	BE	RES	7,(IY+d)	FE	SET	7,(IY+d)
3F			7F			BF			FF		

あとがき

なにやらパズル解きにも似たマシン語プログラミング。十分楽しんでいただけたことでしょう。ただ、5章は少々難解だったでしょうか。X1は表示機能が高く、活用法を詳解するにはかなりの誌面を要するので、本書では資料とサンプル・プログラムにとどめました。また、みなさんから**リクエスト**があれば、グラフィックやPCG、PSGなどをフルにドライブするためのノウハウ書をお届けすることができるでしょう。

X1マシン語プログラミング入門

1984年 9 月 5 日　初版発行
1984年11月30日　第 2 版発行

著　　者　渡辺英行　沼倉 均
発 行 人　塚本慶一郎
発 行 所　株式会社 エム・アイ・エー
〒102 東京都千代田区紀尾井町3-20
電話 (03)265-2461(代)
イラスト　斉藤 修
印刷・製本　豊栄製本株式会社

ISBN4-87170-024-0 C3055 ¥2200E　**定価2200円**